TOMO II

Línea de Investigación

Redes y Sistemas Operativos

Nombre del Egresado

Fausto Stanley Almeida Campos

Nombre del Tema

Implementación de un prototipo de sistema de envío de publicidad vía Bluetooth para un almacén de música, como aplicación del concepto de marketing de proximidad

Número de Proyecto

10

Nombre del Tutor

Ing. David Benavides

Guayaquil, 09 de marzo del 2011

ÍNDICE DEL MANUAL TÉCNICO

1 MODELOS DE PROCESOS 7

1.1 PROCESOS PARA EL MANTENIMIENTO DEL SISTEMA 8

1.2 PROCESOS PARA LA EJECUCIÓN DE REPORTES 10

1.3 PROCESOS PARA EL ENVÍO DE CONTENIDO 12

2 DESCRIPCIÓN DE ESTRUCTURAS DE DATOS 14

2.1 DIAGRAMA DEL MODELO ENTIDAD-RELACIÓN 15

2.2 DICCIONARIO DE DATOS 16

3 MODELOS ORIENTADOS A OBJETOS 25

3.1 DIAGRAMA DE CASOS DE USO 26

3.2 DIAGRAMAS DE CLASES 27

3.2.1 Clases POJOs de entidades. (com.jabluecatch.entidades) 27

3.2.2 Clases POJOs para objetos de transferencia de datos (com.jabluecatch.dto) 28

3.2.3 Clases para objetos de acceso a datos (com.jabluecatch.dao) 29

3.2.4 Clases para objetos de negocio (com.jabluecatch.bo) 31

3.2.5 Clases para objetos de operaciones de comunicación (com.jabluecatch.comm) 33

3.2.6 Clases para objetos de utilería general (com.jabluecatch.util) 35

3.2.7 Clases para objetos de interfaz gráfica de usuario correspondientes al menú de sistema (com.jabluecatch.gui.sistema) 37

3.2.8 Clases para objetos de interfaz gráfica de usuario correspondientes al menú de Campaña (com.jabluecatch.gui.campania) 39

3.2.9 Clases para objetos de interfaz gráfica de usuario correspondientes al menú de Reportes (com.jabluecatch.gui.reportes) 41

3.2.10 Clases para objetos de interfaz gráfica de usuario generales (com.jabluecatch.gui) 43

4 CÓDIGO FUENTE DE LA APLICACIÓN 44

4.1 CLASES E INTERFACES JAVA 45
4.1.1 Paquete com.jabluecatch.entidades 45
4.1.2 Paquete com.jabluecatch.dto 61
4.1.3 Paquete com.jabluecatch.dao 68
4.1.4 Paquete com.jabluecatch.bo 103
4.1.5 Paquete com.jabluecatch.comm 121
4.1.6 Paquete com.jabluecatch.util 135
4.1.7 Paquete com.jabluecatch.gui.sistema 151
4.1.8 Paquete com.jabluecatch.gui.campania 183
4.1.9 Paquete com.jabluecatch.gui.reportes 239
4.1.10 Paquete com.jabluecatch.gui 259
4.1.11 Paquete raiz 279

4.2 ARCHIVOS DE PROPIEDADES 281
4.2.1 Paquete raíz 281

4.3 REPORTES JASPER 282
4.3.1 Paquete com.jabluecatch.recursos.reportes 282

Guayaquil, 09 de marzo del 2011

ÍNDICE DEL MANUAL DE USUARIO

1 INSTALACIÓN 302

1.1 REQUERIMIENTOS RECOMENDADOS DE HARDWARE 303

1.2 REQUERIMIENTOS DE SOFTWARE 303

1.3 PASOS DE INSTALACIÓN 304

2 ORGANIZACIÓN DEL MENU 305

2.1 ROL ADMINISTRADOR 306
2.1.1 Sistema 306
2.1.2 Campaña 306
2.1.3 Reportes 306

2.2 ROL OPERADOR 307
2.2.1 Sistema 307
2.2.2 Campaña 307
2.2.3 Reportes 307

3 OPCIONES DEL SISTEMA 308

3.1 INICIO DEL SISTEMA 309
3.1.1 Login 309

3.2 ROL ADMINISTRADOR 309
3.2.1 Menú: Sistema 309
3.2.2 Menú: Campaña 315
3.2.3 Menú: Reportes 321

3.3 ROL OPERADOR 322
3.3.1 Menú: Sistema 322
3.3.2 Menú: Campaña 323
3.3.3 Menú: Reportes 326

4 REPORTES 331

4.1 REPORTES ADMINISTRATIVOS 332
4.1.1 Reporte de usuarios 332
4.1.2 Reporte de Bitácora de Campaña 333

4.2 REPORTES OPERATIVOS 334
4.2.1 Reporte de Impacto de Campaña 334
4.2.2 Reporte de Costos de Campañas 335

5 ANEXOS 336

5.1 INSTALACIÓN DE JAVA RUNTIME ENVIROMENT 337

5.2 INSTALACIÓN DE BASE DE DATOS MYSQL 339

UNIVERSIDAD DE GUAYAQUIL

FACULTAD DE CIENCIAS MATEMATICAS Y FISICAS

CARRERA DE INGENIERIA EN SISTEMAS COMPUTACIONALES

IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE ENVÍO DE PUBLICIDAD VÍA BLUETOOTH PARA UN ALMACÉN DE MÚSICA, COMO APLICACIÓN DEL CONCEPTO DE MARKETING DE PROXIMIDAD

MANUAL TÉCNICO

AUTOR: FAUSTO STANLEY ALMEIDA CAMPOS

TUTOR: ING. DAVID BENAVIDES

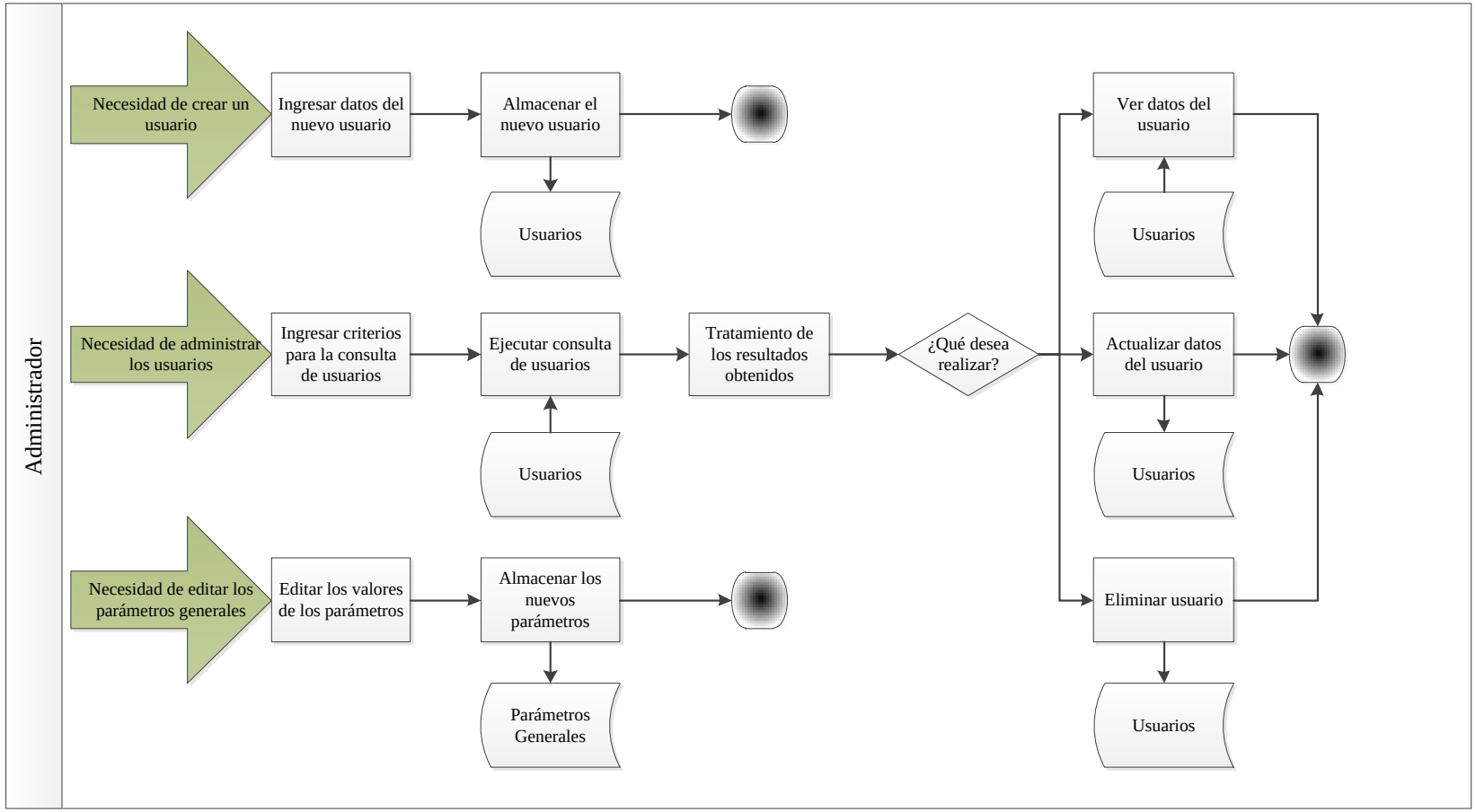
GUAYAQUIL – ECUADOR

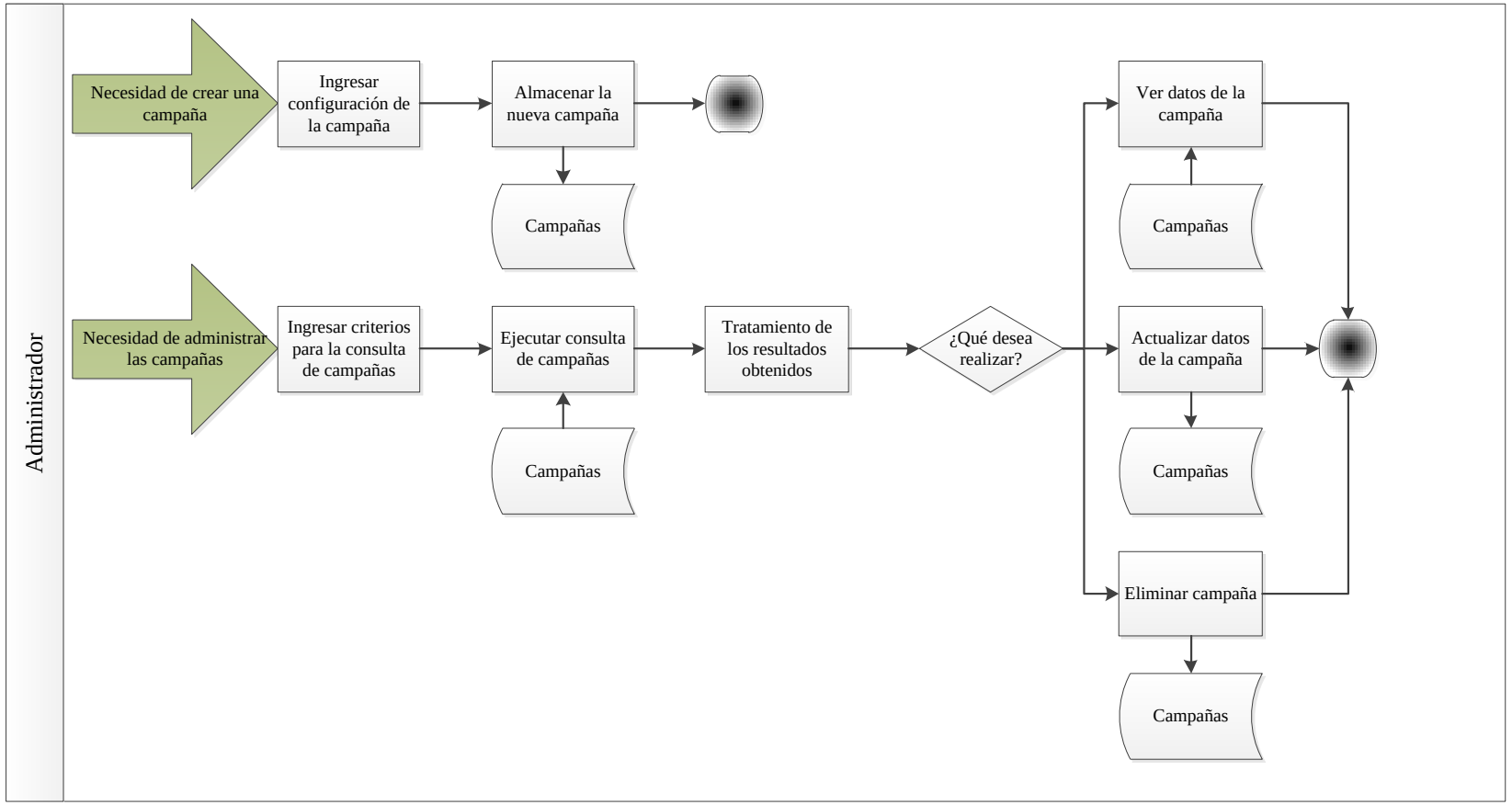
2010

Guayaquil, 16 de agosto del 2010

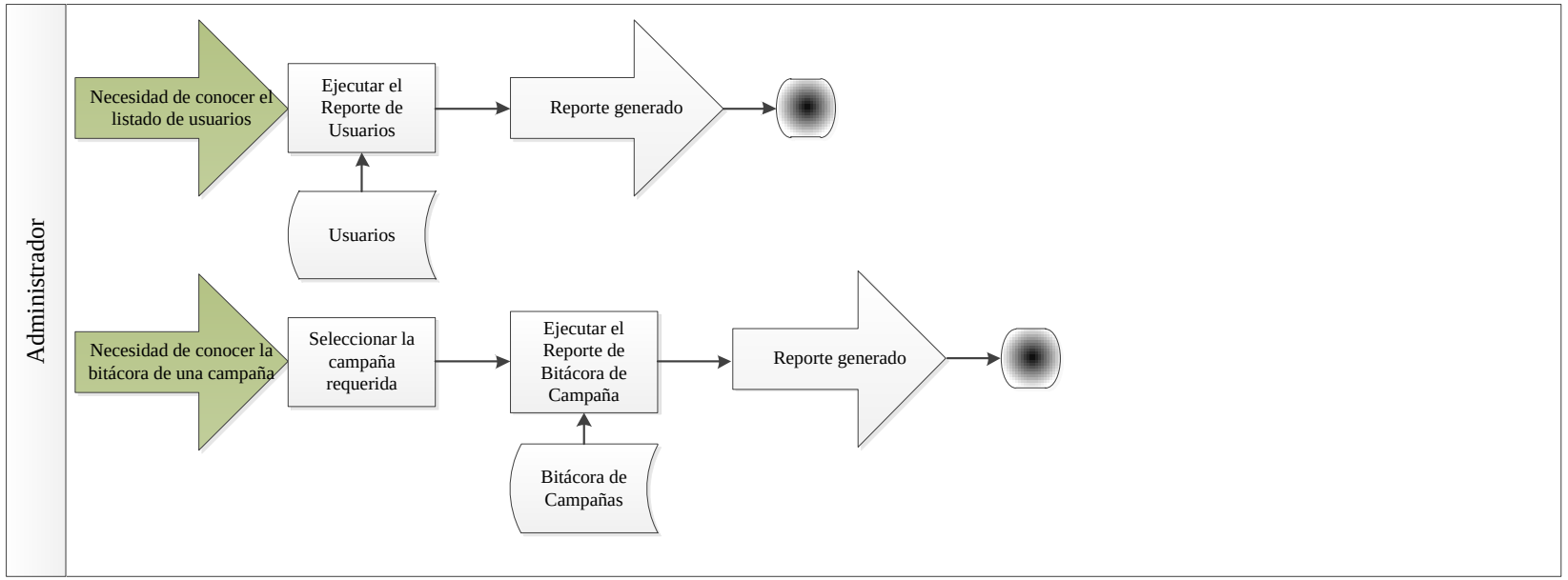
1 MODELOS DE PROCESOS

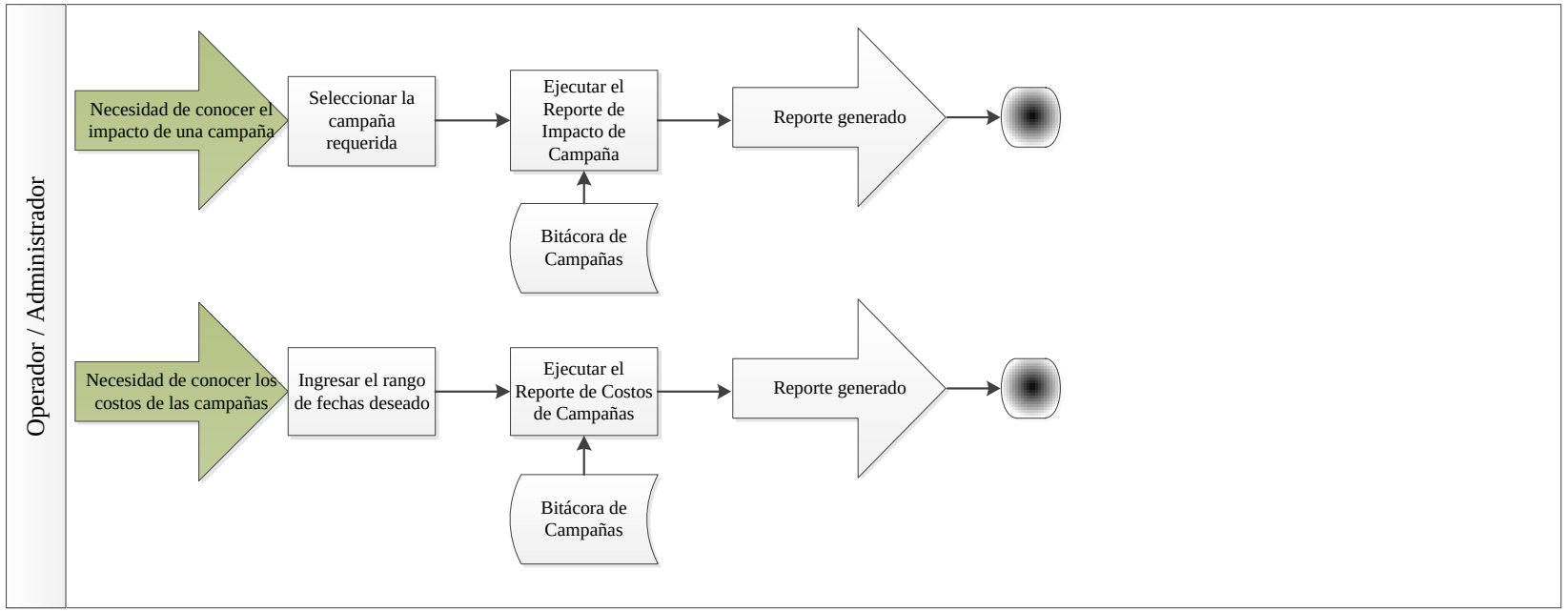
1.1 PROCESOS PARA EL MANTENIMIENTO DEL SISTEMA



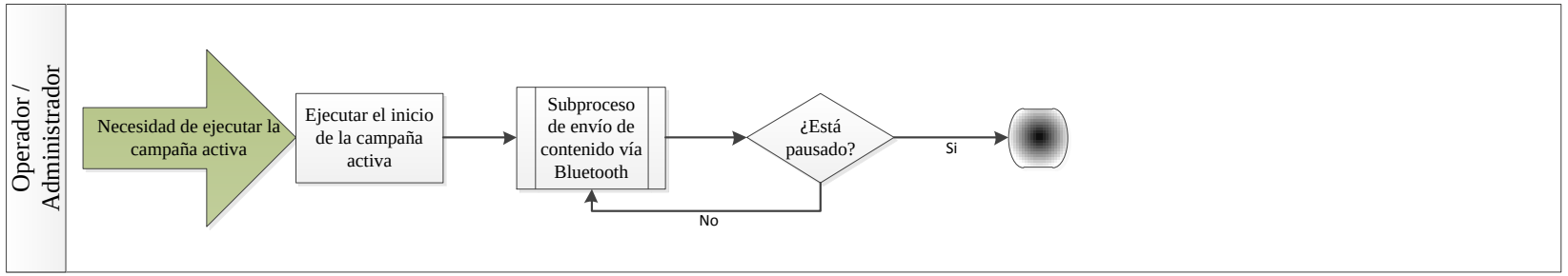

Administrador
Necesidad de crear una campaña
Ingresar configuración de la campaña
Almacenar la nueva campaña
Campañas
Necesidad de administrar las campañas
Ingresar criterios para la consulta de campañas
Ejecutar consulta de campañas
Campañas
Tratamiento de los resultados obtenidos
¿Qué desea realizar?
Ver datos de la campaña
Campañas
Actualizar datos de la campaña
Campañas
Eliminar campaña
Campañas

1.2 PROCESOS PARA LA EJECUCIÓN DE REPORTES

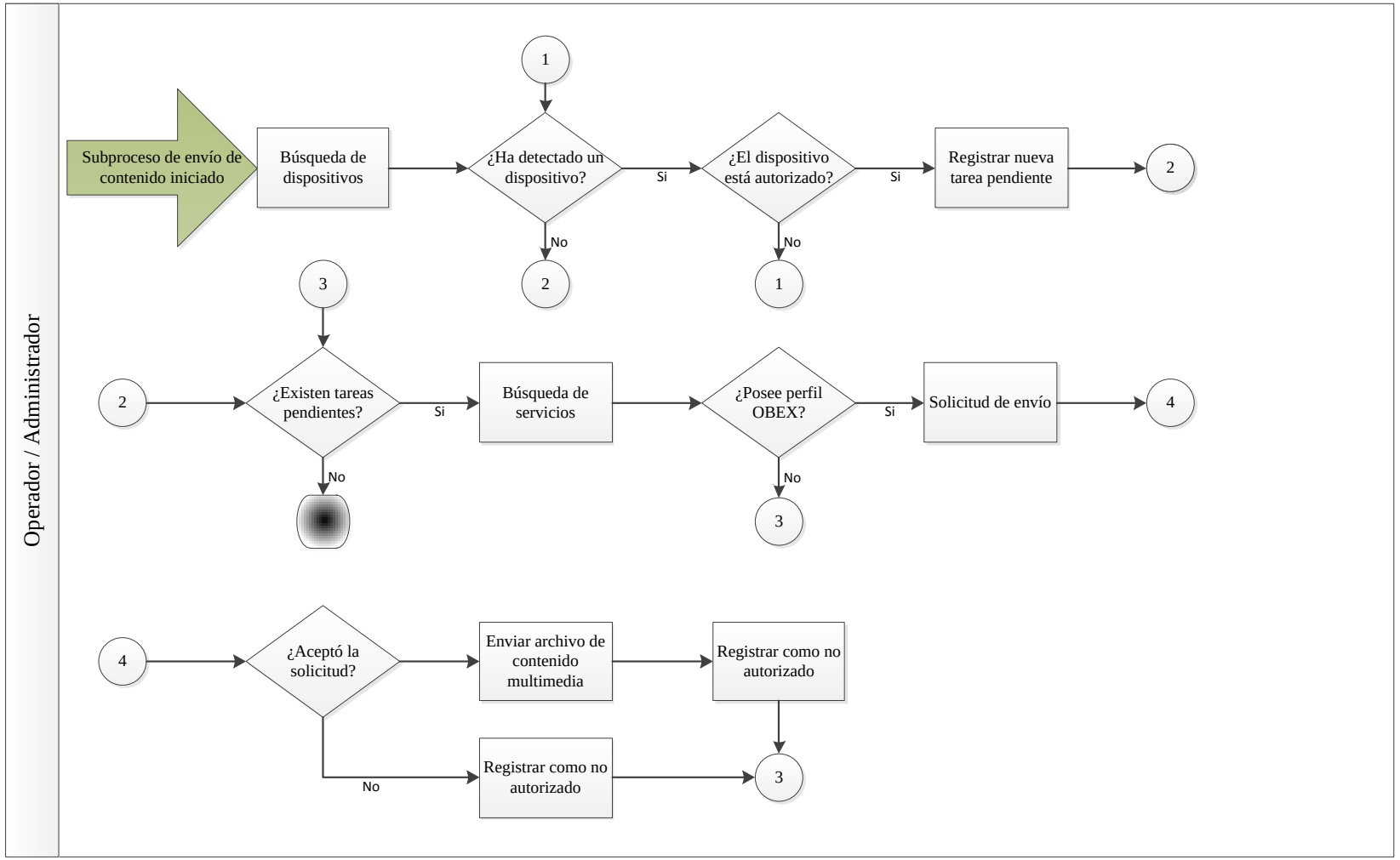

Administrador
Necesidad de conocer el listado de usuarios
Ejecutar el Reporte de Usuarios
Reporte generado
Usuarios
Necesidad de conocer la bitácora de una campaña
Seleccionar la campaña requerida
Ejecutar el Reporte de Bitácora de Campaña
Reporte generado
Bitácora de Campañas


Operador / Administrador
Necesidad de conocer el impacto de una campaña
Seleccionar la campaña requerida
Ejecutar el Reporte de Impacto de Campaña
Reporte generado
Bitácora de Campañas
Necesidad de conocer los costos de las campañas
Ingresar el rango de fechas deseado
Ejecutar el Reporte de Costos de Campañas
Reporte generado
Bitácora de Campañas

1.3 PROCESOS PARA EL ENVÍO DE CONTENIDO

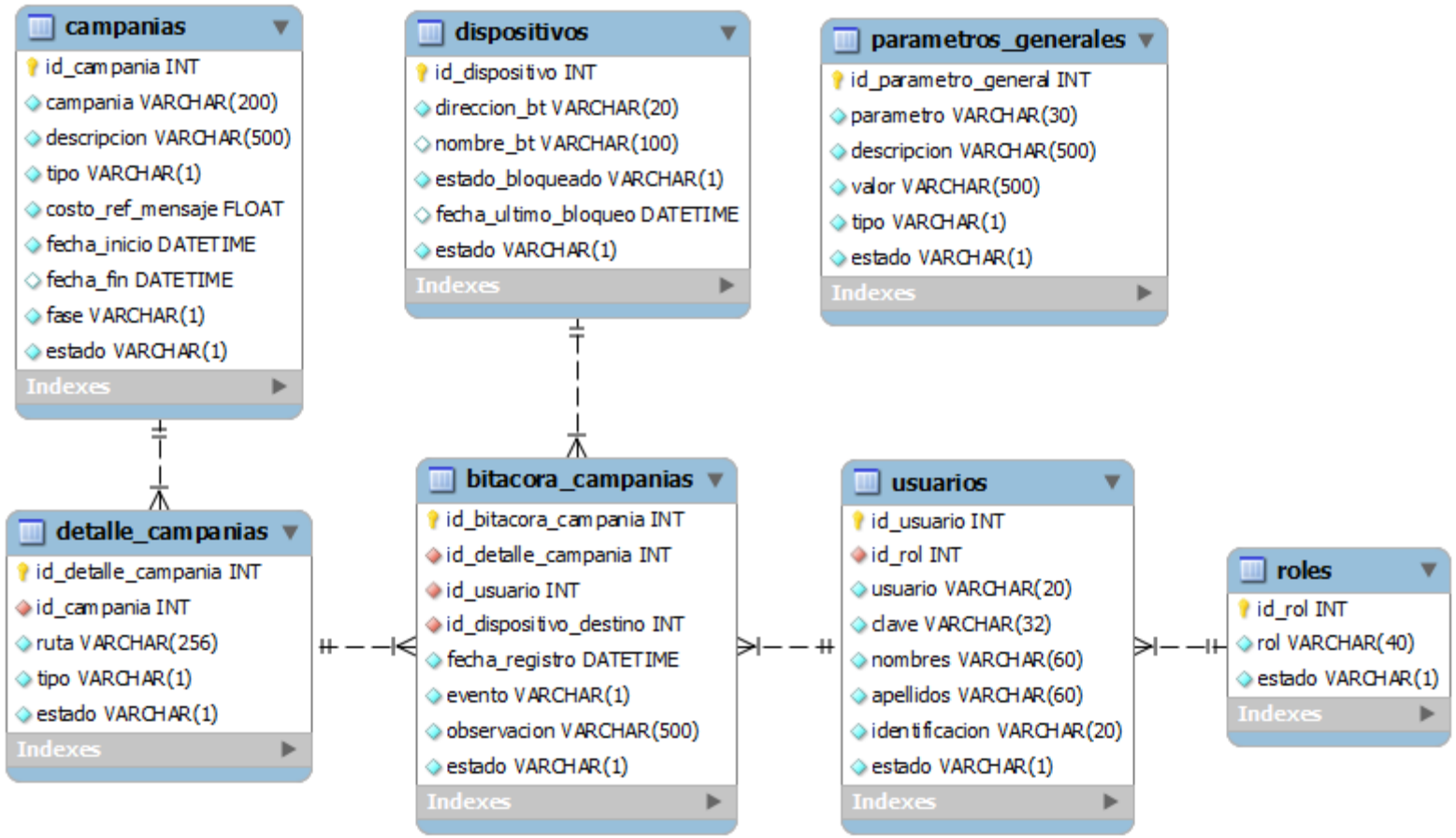


A continuación detallaremos como un subproceso, todas las actividades y evaluación de condiciones que ocurren durante el envío de contenido Bluetooth.


Operador / Administrador
Subproceso de envío de contenido iniciado
Búsqueda de dispositivos
1
¿Ha detectado un dispositivo?
Si
No
2
¿El dispositivo está autorizado?
Si
No
1
Registrar nueva tarea pendiente
2
3
2
¿Existen tareas pendientes?
Si
No
Búsqueda de servicios
¿Posee perfil OBEX?
Si
No
3
Solicitud de envío
4
4
¿Aceptó la solicitud?
Enviar archivo de contenido multimedia
Registrar como no autorizado
No
Registrar como no autorizado
3

2 DESCRIPCIÓN DE ESTRUCTURAS DE DATOS

2.1 DIAGRAMA DEL MODELO ENTIDAD-RELACIÓN



2.2 DICCIONARIO DE DATOS

CAMPANIAS

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_CAMPANIA
CLAVE FORÁNEA	N/A
OBJETIVO	Almacenar la configuración inicial para cada campaña de marketing vía Bluetooth creada.

CAMPO	**TIPO**	**PERMITE NULO**	**OBSERVACION**
ID_CAMPANIA	INT	NO	Identificación única para cada campaña de marketing configurada.
CAMPANIA	VARCHAR(200)	NO	Nombre de la campaña de marketing configurada.
DESCRIPCION	VARCHAR(500)	NO	Descripción detallada o características de la campaña de marketing configurada.
TIPO	VARCHAR(1)	NO	Tipo de campaña de acuerdo a su ejecución. (O=Operacional, P=Permanente).
COSTO_REF_MENSAJE	FLOAT	NO	Precio referencia que tendrá cada mensaje de contenido enviado por el sistema. Necesario para reportes estadísticos.
FECHA_INICIO	DATETIME	NO	Fecha y hora de

			inicio de la campaña de marketing configurada.
FECHA_FIN	DATETIME	SI	Fecha y hora de finalización de la campaña de marketing configurada. Si la campaña es de tipo Permanente, este valor no será necesario.
ESTADO	VARCHAR(1)	NO	Estado del registro de la campaña de marketing. (A=Activa, I=Inactiva).

DETALLE_CAMPANIAS

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_DETALLE_CAMPANIA
CLAVE FORÁNEA	ID_CAMPANIA
OBJETIVO	Almacenar las rutas de los archivos que se enviarán como publicidad durante una determinada campaña de marketing vía Bluetooth.

CAMPO	**TIPO**	**PERMITE NULO**	**OBSERVACION**
ID_DETALLE_CAMPANIA	INT	NO	Identificación única para cada detalle de campaña configurado.
ID_CAMPANIA	INT	NO	Identificación referencial de la campaña de marketing.
RUTA	VARCHAR(256)	NO	Ruta absoluta de

			la ubicación del archivo en el disco duro.
TIPO	VARCHAR(1)	NO	Tipo de archivo. (A=Audio, I=Imagen)
ESTADO	VARCHAR(1)	NO	Estado del registro de detalle de campaña. (A=Activo, I=Inactivo).

DISPOSITIVOS

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_DISPOSITIVO
CLAVE FORÁNEA	N/A
OBJETIVO	Almacenar los dispositivos y sus respectivos estados cada vez que se detecten durante la ejecución de las campañas publicitarias.

CAMPO	**TIPO**	**PERMITE NULO**	**OBSERVACION**
ID_DISPOSITIVO	INT	NO	Identificación única para cada dispositivo configurado en el sistema.
DIRECCION_BT	VARCHAR(40)	NO	Dirección Bluetooth del dispositivo detectado.
NOMBRE_BT	VARCHAR(100)	SI	Nombre que posea el dispositivo detectado.
ESTADO_BLOQUEADO	VARCHAR(1)	NO	Determina si el dispositivo se encuentra o no en un estado bloqueado. (S=Si,

			N=No).
FECHA_ULTIMO_BLOQUEO	DATETIME	SI	Fecha y hora en que el dispositivo fue bloqueado por última vez.
ESTADO	VARCHAR(1)	NO	Estado del dispositivo en el sistema. (A=Activo, I=Inactivo).

BITACORA_CAMPANIAS

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_BITACORA_CAMPANIA
CLAVE FORÁNEA	ID_CAMPANIA, ID_USUARIO
OBJETIVO	Registrar cada uno de los eventos que se generan como producto del proceso de envío de un archivo de publicidad hacia un dispositivo remoto.

CAMPO	**TIPO**	**PERMITE NULO**	**OBSERVACION**
ID_BITACORA_CAMPANIA	INT	NO	Identificación única para cada registro de bitácora generado.
ID_DETALLE_CAMPANIA	INT	NO	Identificación referencial del detalle de campaña de marketing.
ID_USUARIO	INT	NO	Identificación referencial del usuario conectado al sistema.
ID_DISPOSITIVO_DESTINO	INT	NO	Identificación referencial al dispositivo.
FECHA_REGISTRO	DATETIME	NO	Fecha y hora en que se registra el

			evento.
EVENTO	VARCHAR(1)	NO	Evento registrado. (I=Inicio de transmisión, E=Transmisión exitosa, X=Transmisión fallida, C=Transmisión cancelada por el cliente)
OBSERVACIÓN	VARCHAR(500)	NO	Cualquier novedad que ocurra durante el proceso correspondiente.
ESTADO	VARCHAR(1)	NO	Estado del registro de bitácora. (A=Activo, I=Inactivo).

USUARIOS

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_USUARIO
CLAVE FORÁNEA	ID_ROL
OBJETIVO	Almacenar la información de todos los usuarios administradores y operadores configurados en el sistema.

CAMPO	**TIPO**	**PERMITE NULO**	**OBSERVACION**
ID_USUARIO	INT	NO	Identificación única para cada usuario configurado en el sistema.
ID_ROL	INT	NO	Identificación referencial del rol que posee el

			usuario.
USUARIO	VARCHAR(20)	NO	Nombre del usuario configurado.
CLAVE	VARCHAR(32)	NO	Contraseña para inicio de sesión para el usuario.
NOMBRES	VARCHAR(60)	NO	Nombres de la persona.
APELLIDOS	VARCHAR(60)		Apellidos de la persona.
IDENTIFICACION	VARCHAR(20)	NO	Identificación de la persona. Cédula.
ESTADO	VARCHAR(1)	NO	Estado del registro de usuario de sistema. (A=Activo, I=Inactivo).

ROLES

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_ROL
CLAVE FORÁNEA	N/A
OBJETIVO	Almacenar la información de todos los roles configurados en el sistema.

CAMPO	**TIPO**	**PERMITE NULO**	**OBSERVACION**
ID_ROL	INT	NO	Identificación única para cada rol configurado en el sistema.
ROL	VARCHAR(40)	NO	Nombre del rol configurado.
ESTADO	VARCHAR(1)	NO	Estado del registro de rol de sistema. (A=Activo, I=Inactivo).

PARAMETROS_GENERALES

ESQUEMA	BLUECATCH
CLAVE PRIMARIA	ID_PARAMETRO_GENERAL
CLAVE FORÁNEA	N/A
OBJETIVO	Almacenar temporalmente los dispositivos que hayan pasado a un estado de bloqueo o restricción hasta que el sistema se encargue de desbloquearlos.

CAMPO	TIPO	PERMITE NULO	OBSERVACION
ID_PARAMETRO_GENERAL	INT	NO	Identificación única para cada parámetro configurado en el sistema.
PARAMETRO	VARCHAR(30)	NO	Nombre del parámetro general configurado.
DESCRIPCION	VARCHAR(500)	NO	Descripción u objetivo del parámetro general configurado.
VALOR	VARCHAR(500)	NO	Valor del parámetro generado configurado.
TIPO	VARCHAR(1)	NO	Tipo del parámetro configurado. (V=Valor, M=Mensaje).
ESTADO	VARCHAR(1)	NO	Estado del registro de parámetro de sistema. (A=Activo, I=Inactivo).

2.3 ESPACIO EN DISCO NECESARIO

Tomando en cuenta las tablas principales que intervienen en el proceso de creación, modificación y ejecución de las campañas, se ha realizado una proyección para calcular el espacio necesario en disco con el que s edeberá contar asumiendo que cada uno de los campos de las tablas mencionadas esté lleno en su totalidad y que en la ejecución de dicha campaña se cuente con las cantidades de dispositivos y eventos referenciales aquí tomados en cuenta.

1 registro de campaña:

ID_CAMPANIA	INT	4
CAMPANIA	VARCHAR(200)	202
DESCRIPCION	VARCHAR(500)	502
TIPO	VARCHAR(1)	3
COSTO_REF_MENSAJE	FLOAT	4
FECHA_INICIO	DATETIME	8
FECHA_FIN	DATETIME	8
ESTADO	VARCHAR(1)	3
		734 bytes

6 registros de detalles:

ID_DETALLE_CAMPANIA	INT	4
ID_CAMPANIA	INT	4
RUTA	VARCHAR(256)	258
TIPO	VARCHAR(1)	3
ESTADO	VARCHAR(1)	3
	6 x	272 bytes
		1632 bytes

100 resgistros de dispositivos:

ID_DISPOSITIVO	INT	4	
DIRECCION_BT	VARCHAR(40)	42	
NOMBRE_BT	VARCHAR(100)	102	
ESTADO_BLOQUEADO	VARCHAR(1)	3	
FECHA_ULTIMO_BLOQUEO	DATETIME	8	
ESTADO	VARCHAR(1)	3	
	100 x	162	bytes
		16200	**bytes**

600 resgistros de bitacora:

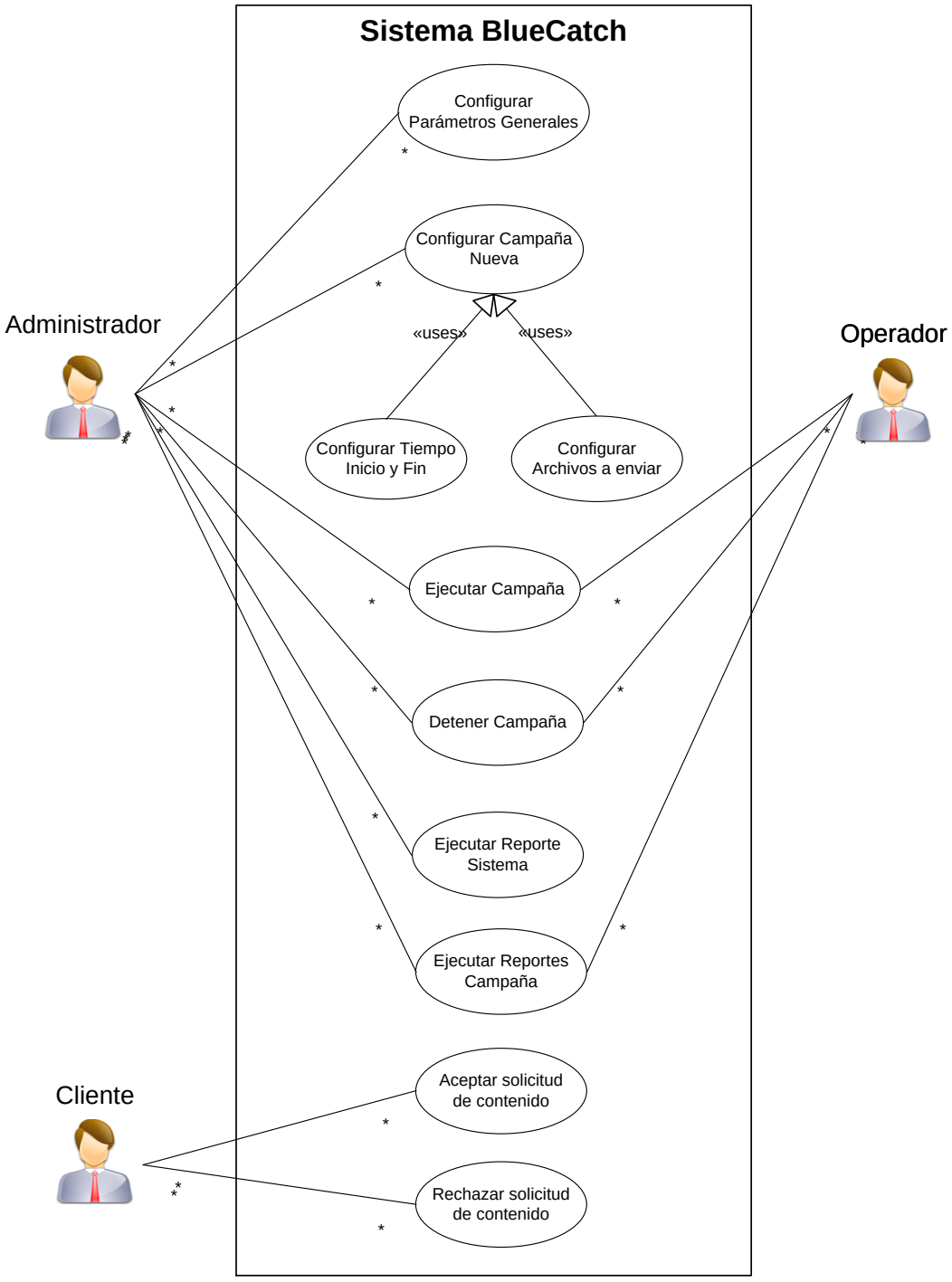
ID_BITACORA_CAMPANIA	INT	4	
ID_DETALLE_CAMPANIA	INT	4	
ID_USUARIO	INT	4	
ID_DISPOSITIVO_DESTINO	INT	4	
FECHA_REGISTRO	DATETIME	8	
EVENTO	VARCHAR(1)	3	
OBSERVACIÓN	VARCHAR(500)	502	
ESTADO	VARCHAR(1)	3	
	600 x	532	bytes
		319200	**bytes**

Total por campaña:	337766	bytes
	329,85	**kilobytes**

Como nos podemos dar cuenta, con los valores hipotéticos planteados y basados en la cantidad de memoria en bytes que ocupa cada campo de las tablas mencionadas, podemos concluir que cada campaña que encuentre unos 100 dispositivos y a cada uno le realice dos ciclos de transferencia de datos, ocupara 330 kilobytes en disco duro aproximadamente.

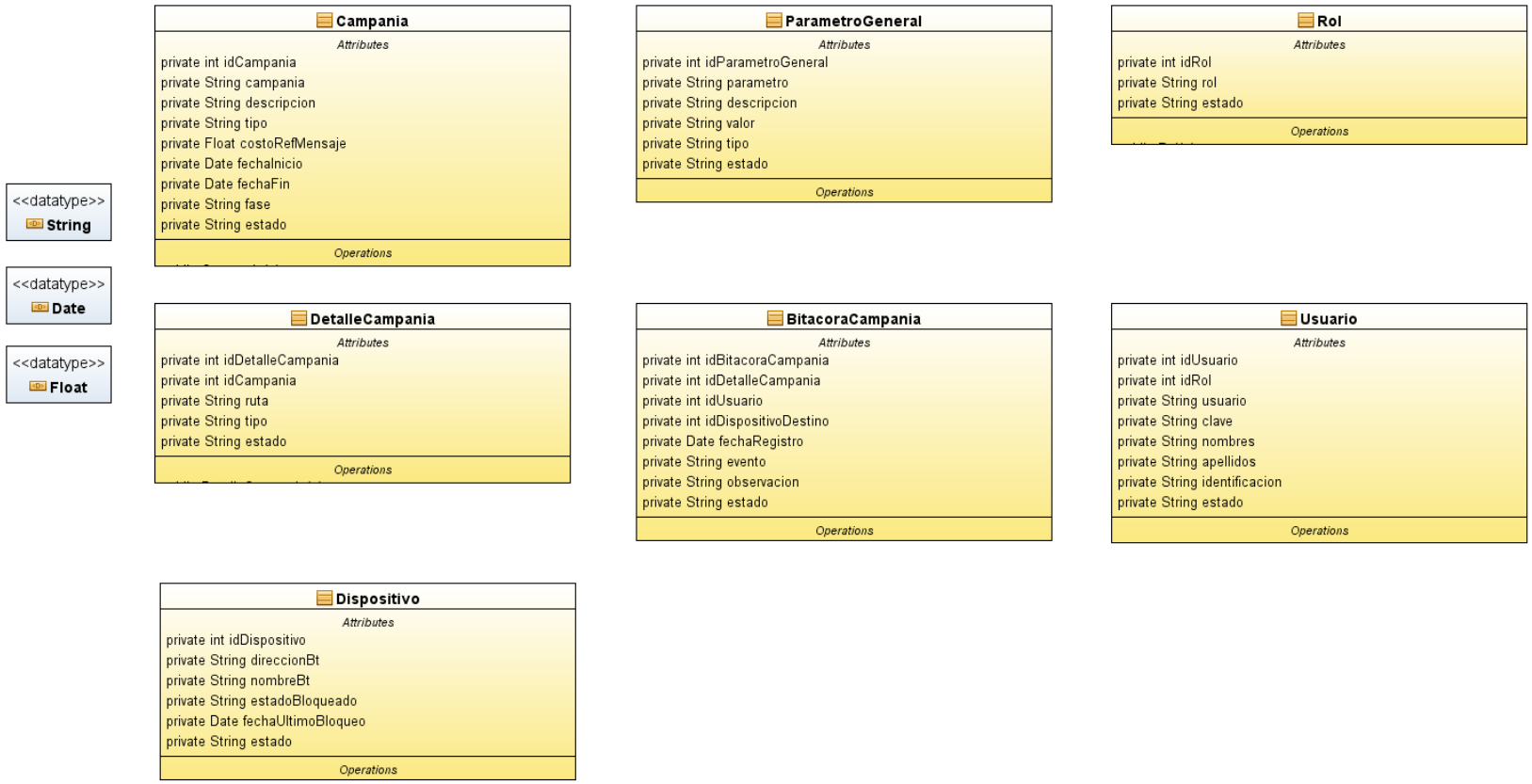
3 MODELOS ORIENTADOS A OBJETOS

3.1 DIAGRAMA DE CASOS DE USO

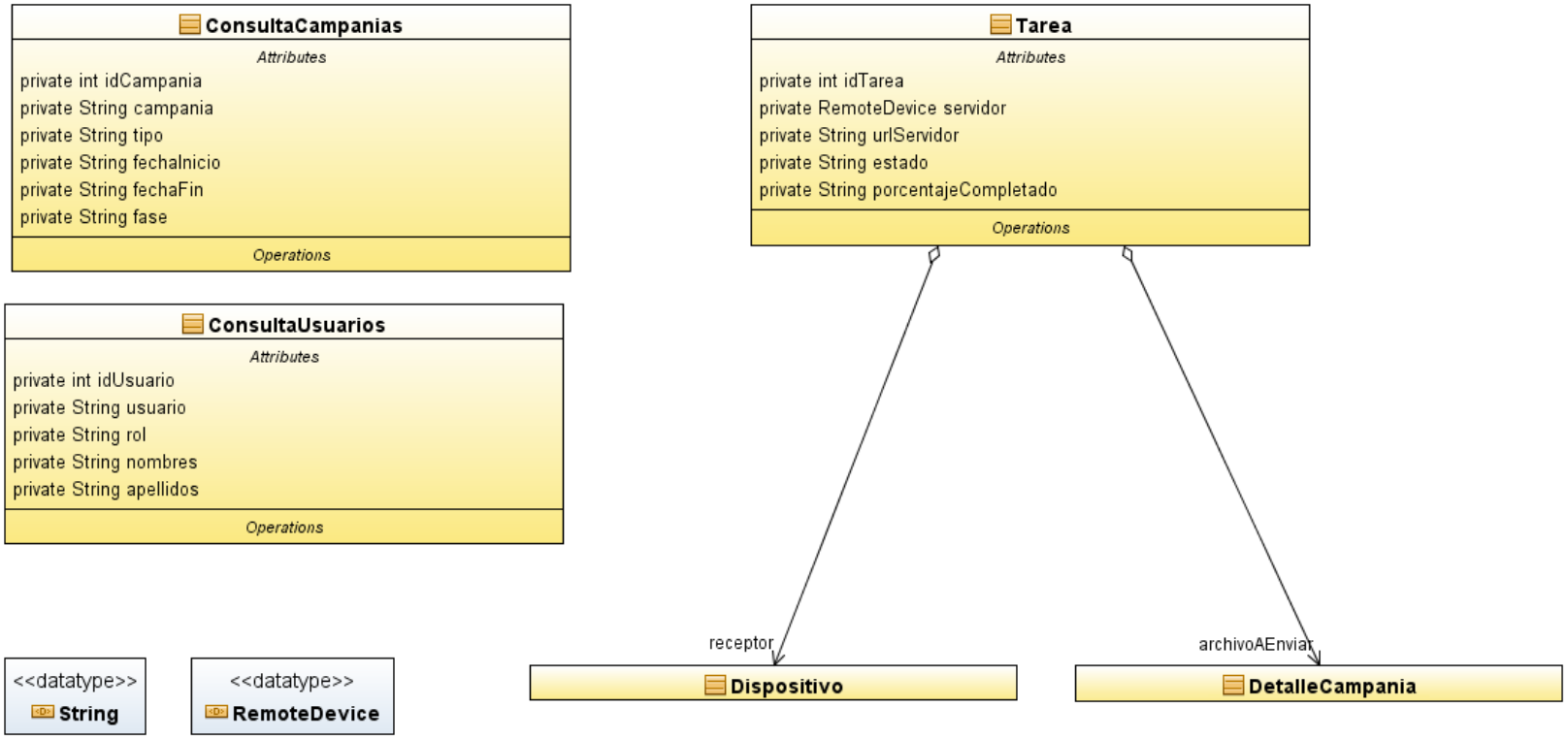


3.2 DIAGRAMAS DE CLASES

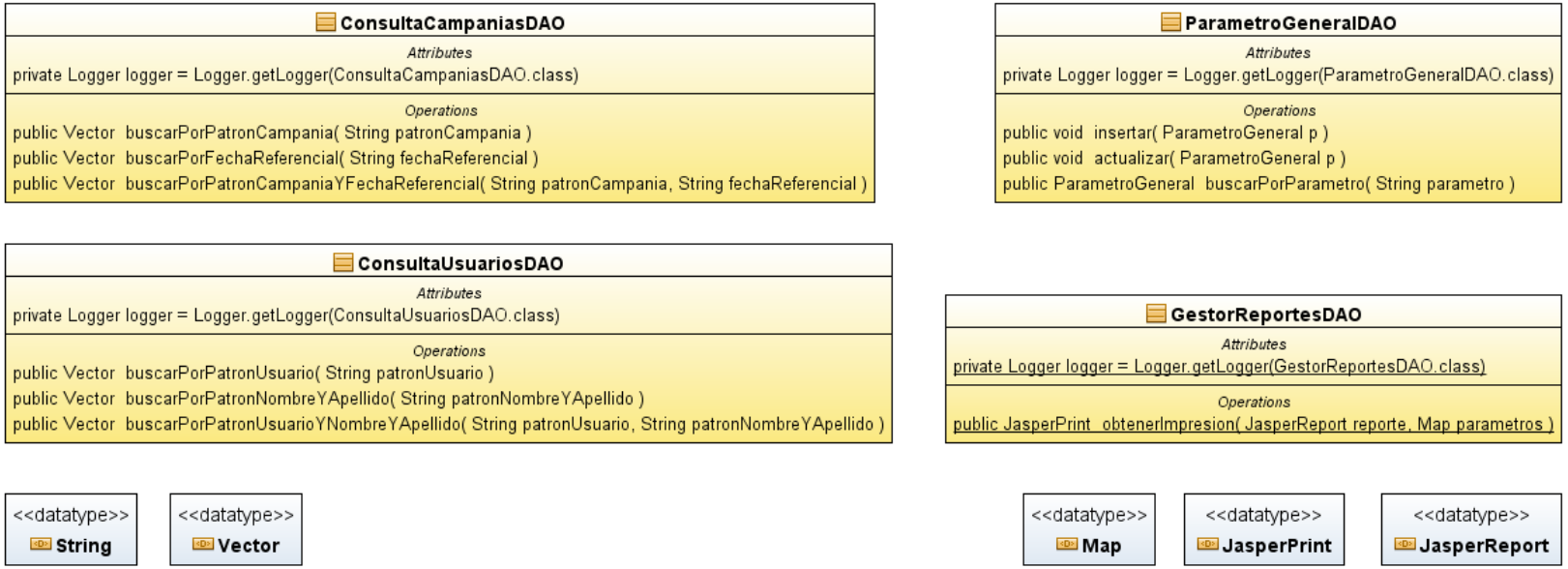
3.2.1 Clases POJOs de entidades. (com.jabluecatch.entidades)

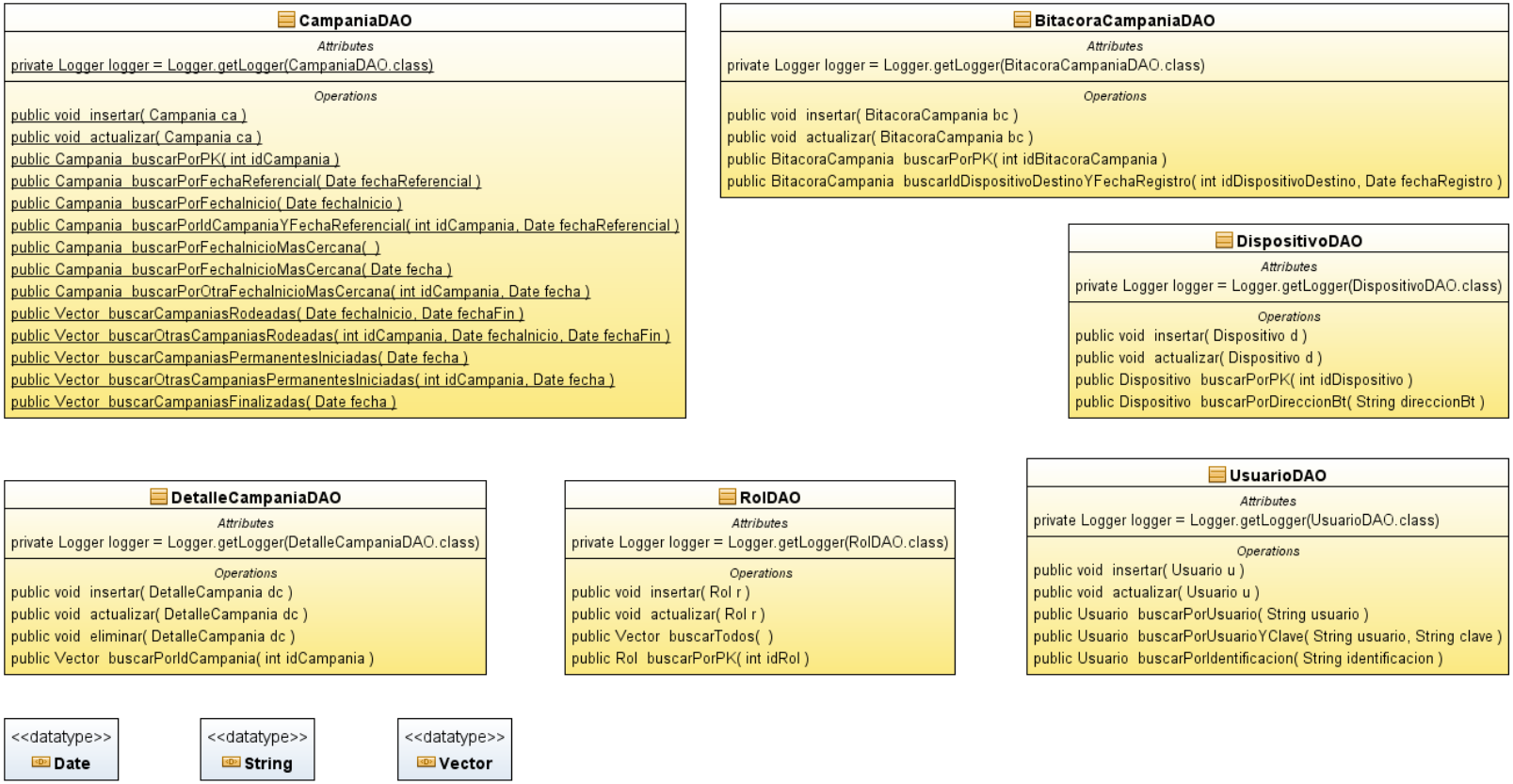


3.2.2 Clases POJOs para objetos de transferencia de datos (com.jabluecatch.dto)

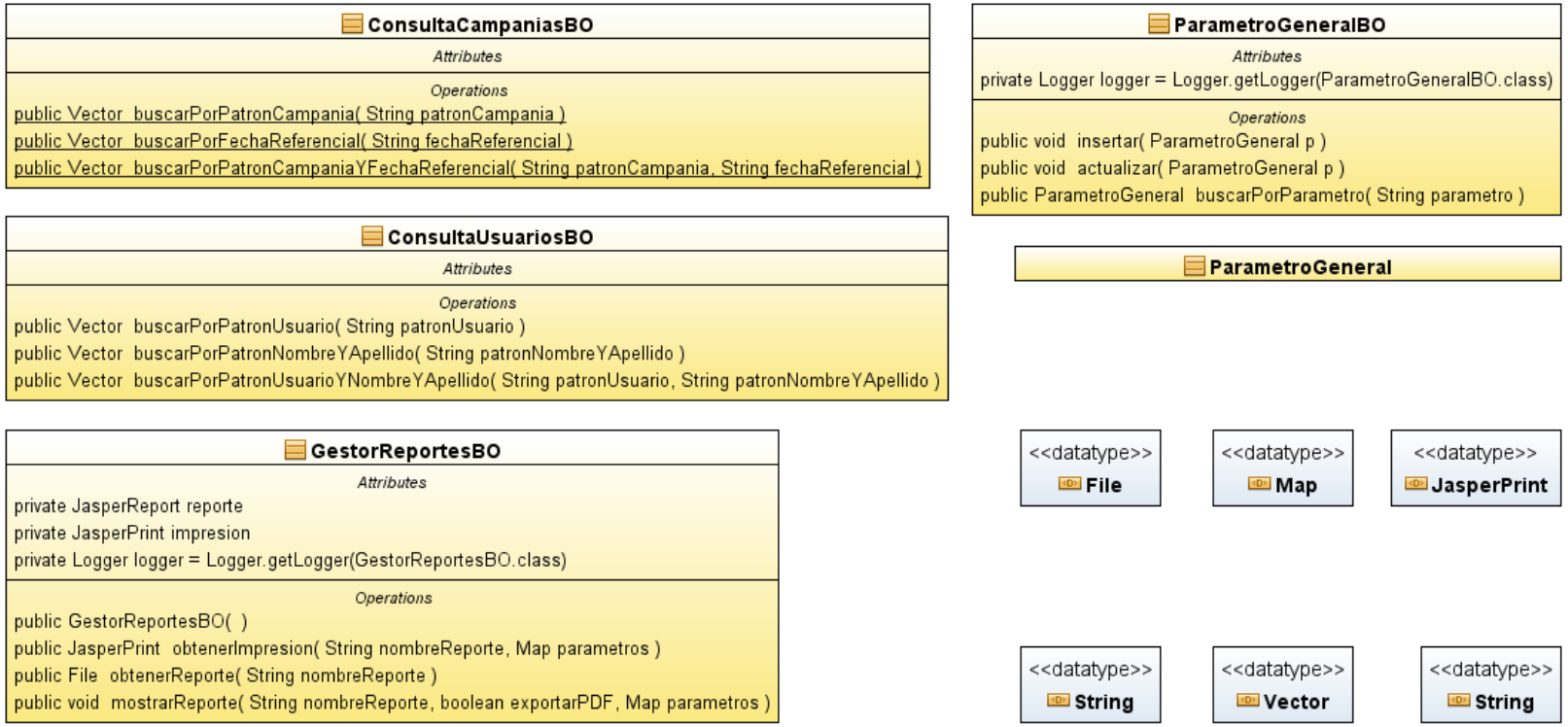


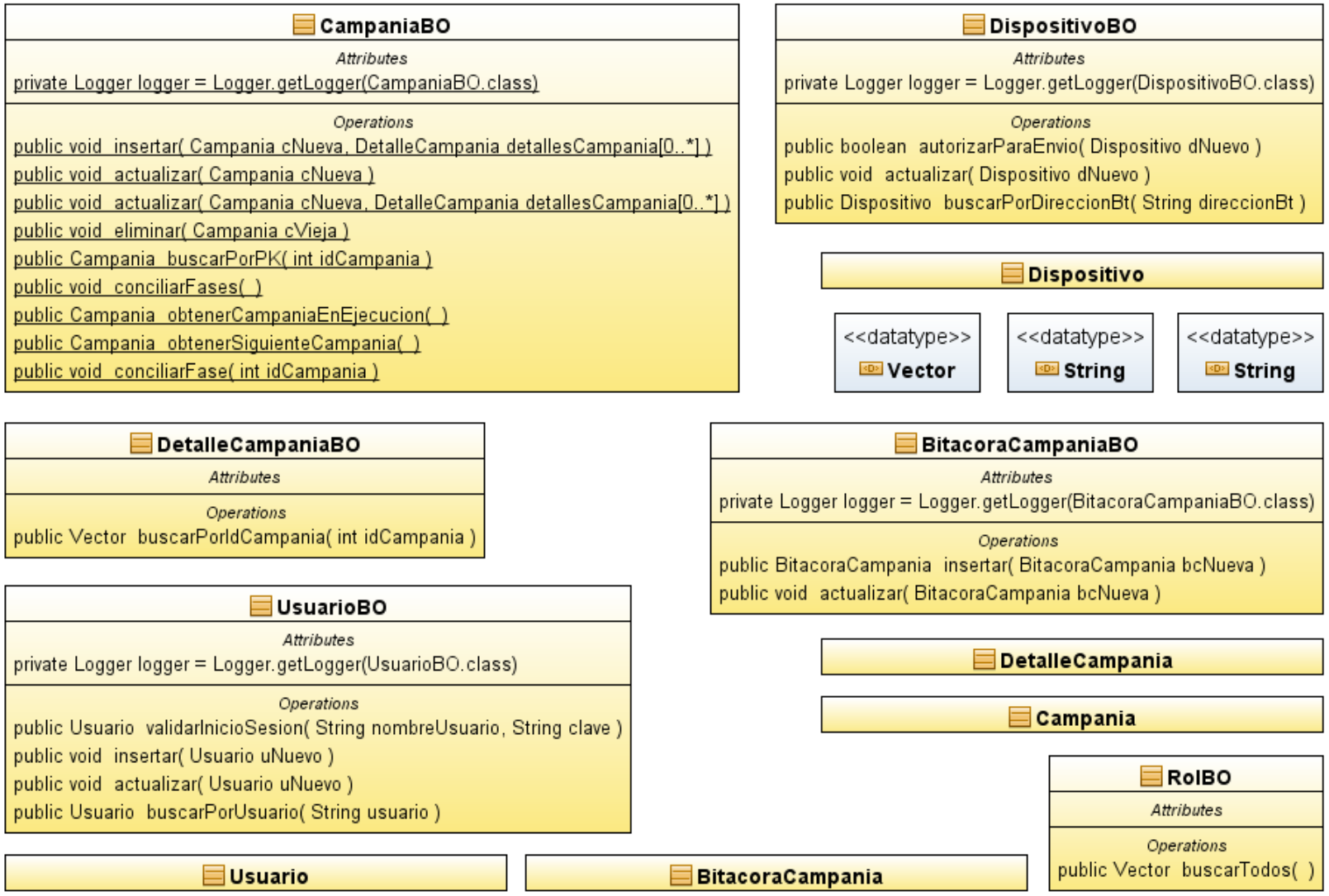
3.2.3 Clases para objetos de acceso a datos (com.jabluecatch.dao)



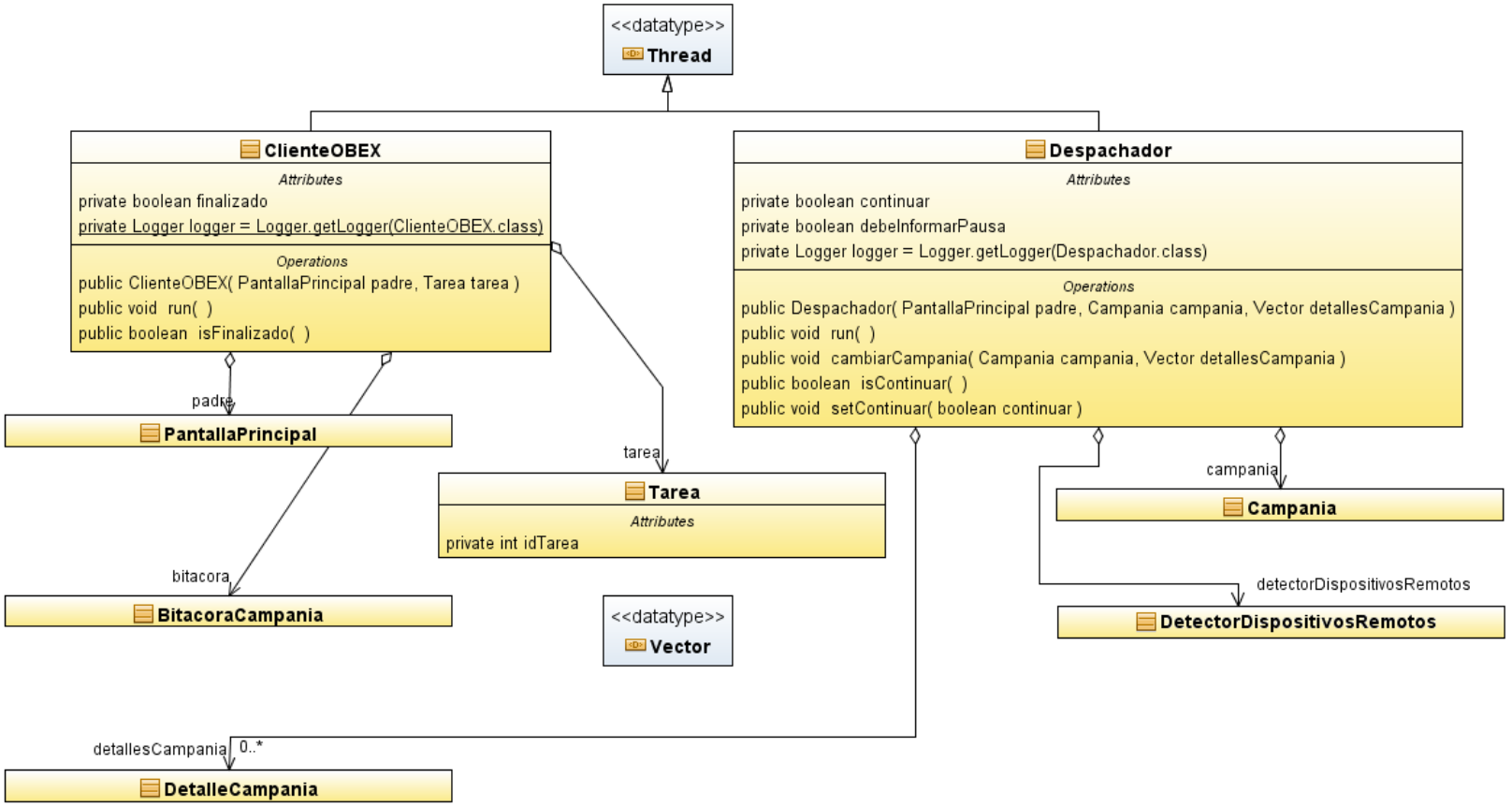

CampaniaDAO
Attributes
private Logger logger = Logger.getLogger(CampaniaDAO.class)
Operations
public void insertar(Campania ca)
public void actualizar(Campania ca)
public Campania buscarPorPK(int idCampania)
public Campania buscarPorFechaReferencial(Date fechaReferencial)
public Campania buscarPorFechaInicio(Date fechaInicio)
public Campania buscarPorIdCampaniaYFechaReferencial(int idCampania, Date fechaReferencial)
public Campania buscarPorFechaInicioMasCercana( )
public Campania buscarPorFechaInicioMasCercana(Date fecha)
public Campania buscarPorOtraFechaInicioMasCercana(int idCampania, Date fecha)
public Vector buscarCampaniasRodeadas(Date fechaInicio, Date fechaFin)
public Vector buscarOtrasCampaniasRodeadas(int idCampania, Date fechaInicio, Date fechaFin)
public Vector buscarCampaniasPermanentesIniciadas(Date fecha)
public Vector buscarOtrasCampaniasPermanentesIniciadas(int idCampania, Date fecha)
public Vector buscarCampaniasFinalizadas(Date fecha)
BitacoraCampaniaDAO
Attributes
private Logger logger = Logger.getLogger(BitacoraCampaniaDAO.class)
Operations
public void insertar(BitacoraCampania bc)
public void actualizar(BitacoraCampania bc)
public BitacoraCampania buscarPorPK(int idBitacoraCampania)
public BitacoraCampania buscarIdDispositivoDestinoYFechaRegistro(int idDispositivoDestino, Date fechaRegistro)
DispositivoDAO
Attributes
private Logger logger = Logger.getLogger(DispositivoDAO.class)
Operations
public void insertar(Dispositivo d)
public void actualizar(Dispositivo d)
public Dispositivo buscarPorPK(int idDispositivo)
public Dispositivo buscarPorDireccionBt(String direccionBt)
DetalleCampaniaDAO
Attributes
private Logger logger = Logger.getLogger(DetalleCampaniaDAO.class)
Operations
public void insertar(DetalleCampania dc)
public void actualizar(DetalleCampania dc)
public void eliminar(DetalleCampania dc)
public Vector buscarPorIdCampania(int idCampania)
RolDAO
Attributes
private Logger logger = Logger.getLogger(RolDAO.class)
Operations
public void insertar(Rol r)
public void actualizar(Rol r)
public Vector buscarTodos( )
public Rol buscarPorPK(int idRol)
UsuarioDAO
Attributes
private Logger logger = Logger.getLogger(UsuarioDAO.class)
Operations
public void insertar(Usuario u)
public void actualizar(Usuario u)
public Usuario buscarPorUsuario(String usuario)
public Usuario buscarPorUsuarioYClave(String usuario, String clave)
public Usuario buscarPorIdentificacion(String identificacion)
<<datatype>>
Date
<<datatype>>
String
<<datatype>>
Vector

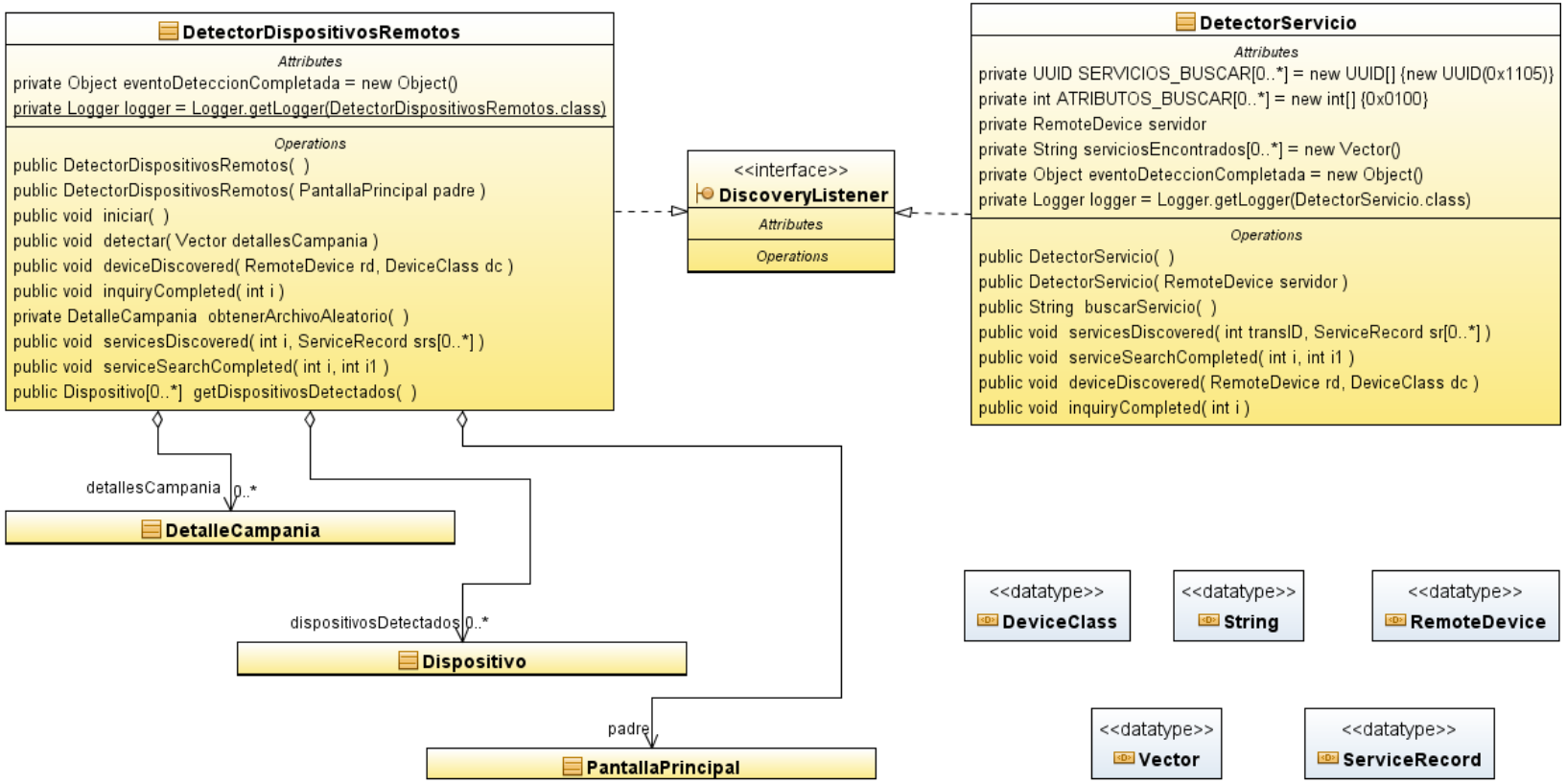
3.2.4 Clases para objetos de negocio (com.jabluecatch.bo)



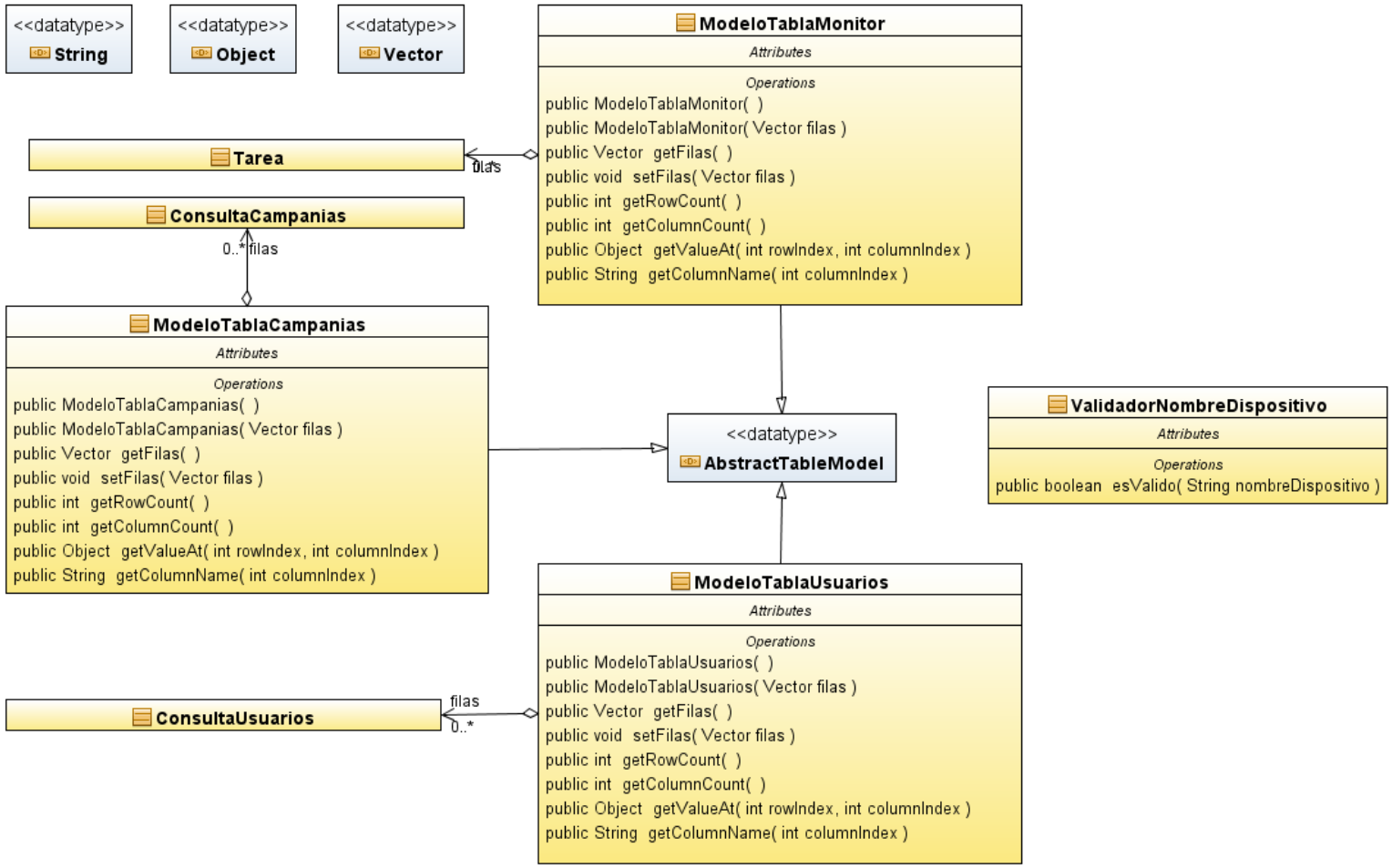

CampaniaBO
Attributes
private Logger logger = Logger.getLogger(CampaniaBO.class)
Operations
public void insertar(Campania cNueva, DetalleCampania detallesCampania[0..*])
public void actualizar(Campania cNueva)
public void actualizar(Campania cNueva, DetalleCampania detallesCampania[0..*])
public void eliminar(Campania cVieja)
public Campania buscarPorPK(int idCampania)
public void conciliarFases( )
public Campania obtenerCampaniaEnEjecucion( )
public Campania obtenerSiguienteCampania( )
public void conciliarFase(int idCampania)
DispositivoBO
Attributes
private Logger logger = Logger.getLogger(DispositivoBO.class)
Operations
public boolean autorizarParaEnvio(Dispositivo dNuevo)
public void actualizar(Dispositivo dNuevo)
public Dispositivo buscarPorDireccionBt(String direccionBt)
Dispositivo
<<datatype>>
Vector
<<datatype>>
String
<<datatype>>
String
DetalleCampaniaBO
Attributes
Operations
public Vector buscarPorIdCampania(int idCampania)
BitacoraCampaniaBO
Attributes
private Logger logger = Logger.getLogger(BitacoraCampaniaBO.class)
Operations
public BitacoraCampania insertar(BitacoraCampania bcNueva)
public void actualizar(BitacoraCampania bcNueva)
UsuarioBO
Attributes
private Logger logger = Logger.getLogger(UsuarioBO.class)
Operations
public Usuario validarInicioSesion(String nombreUsuario, String clave)
public void insertar(Usuario uNuevo)
public void actualizar(Usuario uNuevo)
public Usuario buscarPorUsuario(String usuario)
DetalleCampania
Campania
RolBO
Attributes
Operations
public Vector buscarTodos( )
Usuario
BitacoraCampania

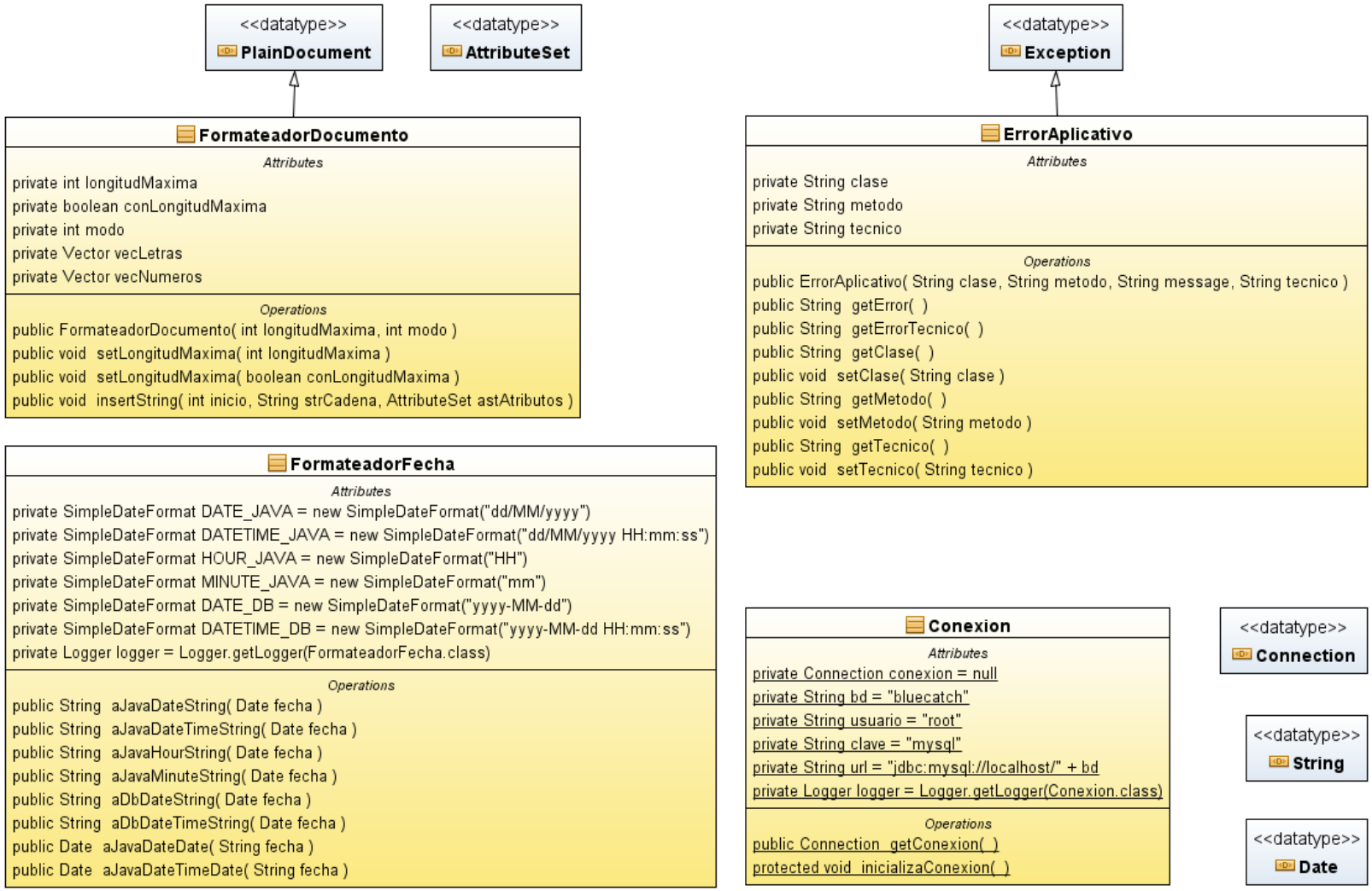
3.2.5 Clases para objetos de operaciones de comunicación (com.jabluecatch.comm)



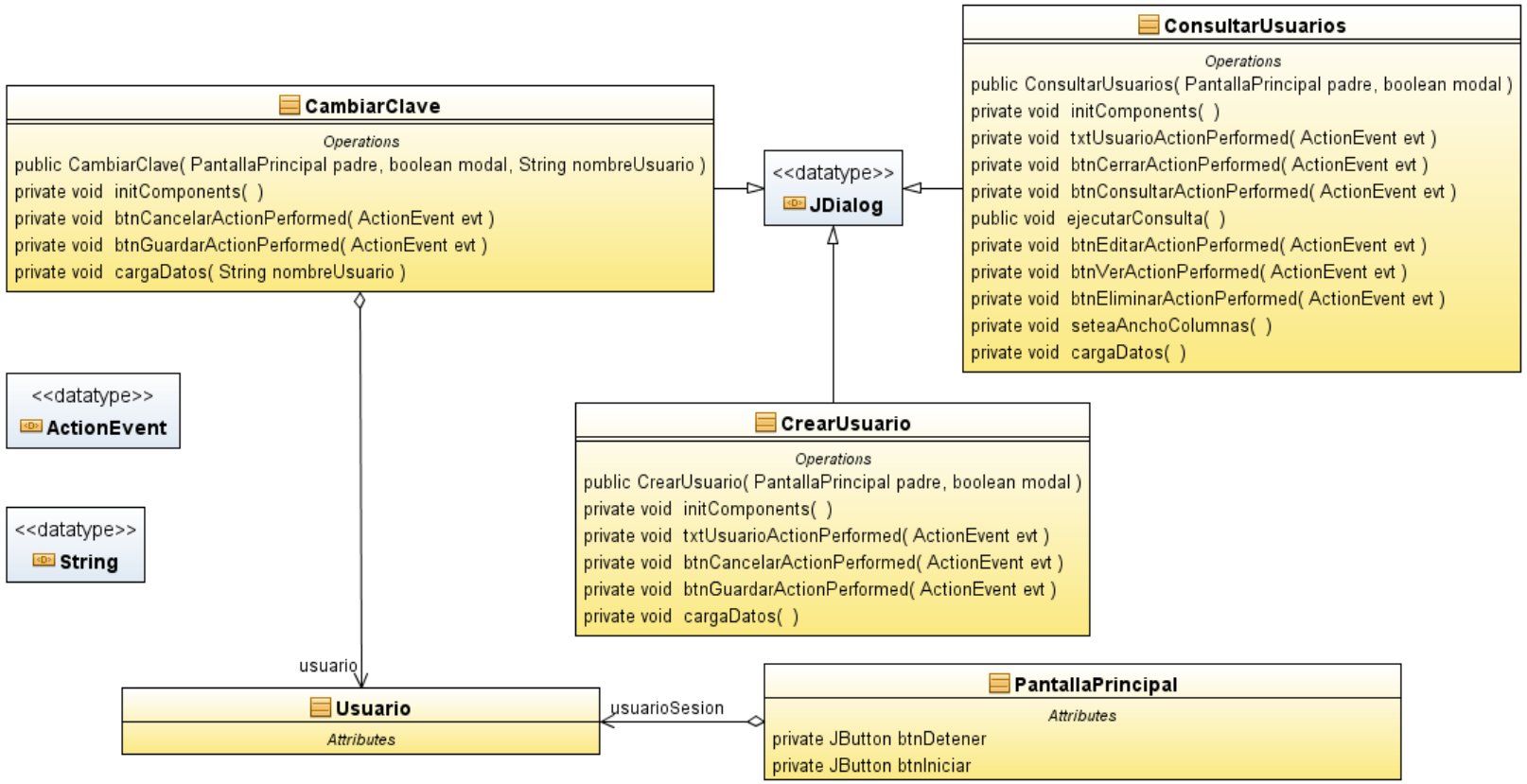

DetectorDispositivosRemotos
Attributes
private Object eventoDeteccionCompletada = new Object()
private Logger logger = Logger.getLogger(DetectorDispositivosRemotos.class)
Operations
public DetectorDispositivosRemotos( )
public DetectorDispositivosRemotos(PantallaPrincipal padre)
public void iniciar( )
public void detectar(Vector detallesCampania)
public void deviceDiscovered(RemoteDevice rd, DeviceClass dc)
public void inquiryCompleted(int i)
private DetalleCampania obtenerArchivoAleatorio( )
public void servicesDiscovered(int i, ServiceRecord srs[0..*])
public void serviceSearchCompleted(int i, int i1)
public Dispositivo[0..*] getDispositivosDetectados( )
<<interface>>
DiscoveryListener
Attributes
Operations
DetectorServicio
Attributes
private UUID SERVICIOS_BUSCAR[0..*] = new UUID[] {new UUID(0x1105)}
private int ATRIBUTOS_BUSCAR[0..*] = new int[] {0x0100}
private RemoteDevice servidor
private String serviciosEncontrados[0..*] = new Vector()
private Object eventoDeteccionCompletada = new Object()
private Logger logger = Logger.getLogger(DetectorServicio.class)
Operations
public DetectorServicio( )
public DetectorServicio(RemoteDevice servidor)
public String buscarServicio( )
public void servicesDiscovered(int transID, ServiceRecord sr[0..*])
public void serviceSearchCompleted(int i, int i1)
public void deviceDiscovered(RemoteDevice rd, DeviceClass dc)
public void inquiryCompleted(int i)
detallesCampania
0..*
DetalleCampania
dispositivosDetectados
0..*
Dispositivo
padre
PantallaPrincipal
<<datatype>>
DeviceClass
<<datatype>>
String
<<datatype>>
RemoteDevice
<<datatype>>
Vector
<<datatype>>
ServiceRecord

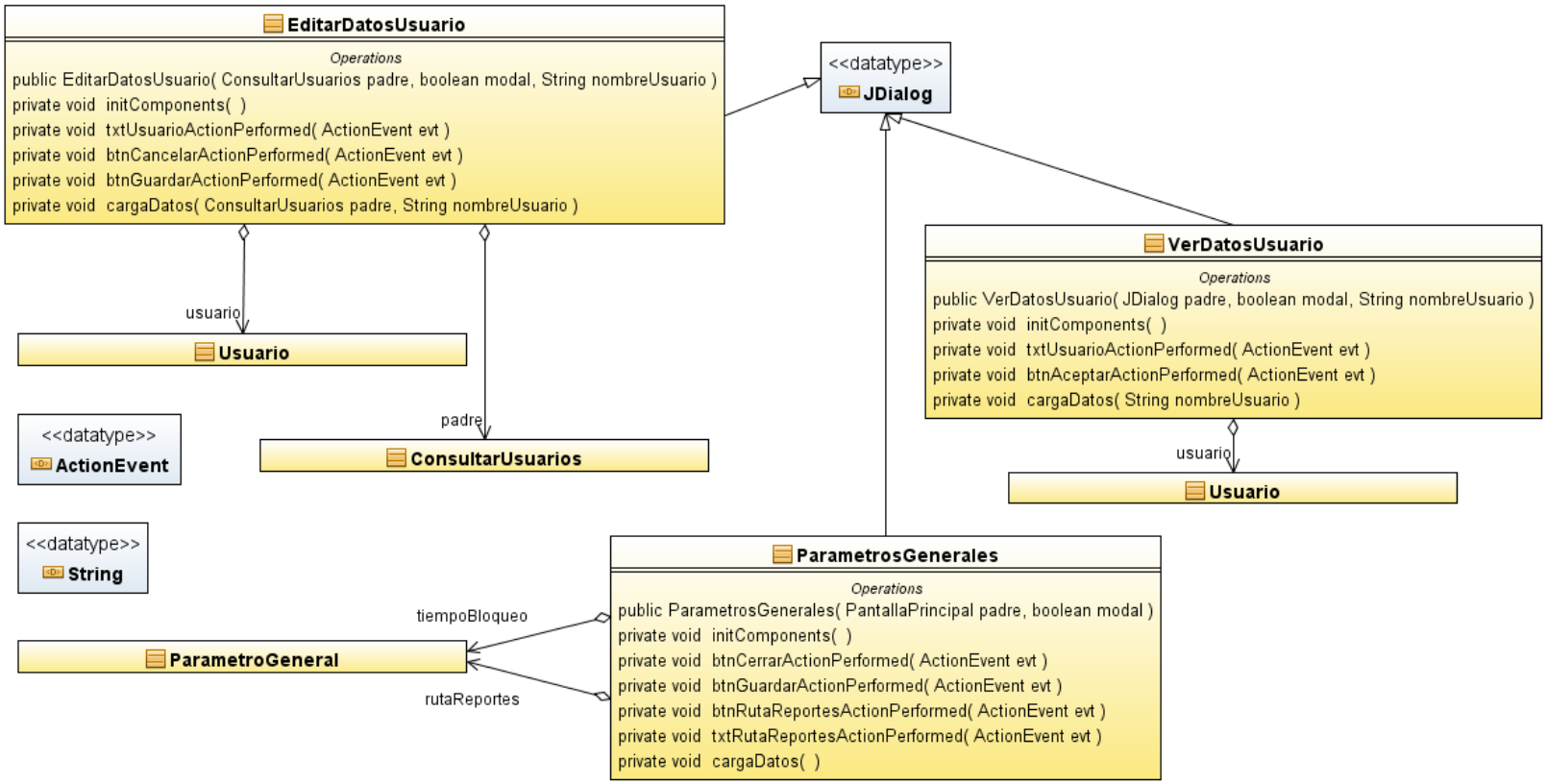
3.2.6 Clases para objetos de utilería general (com.jabluecatch.util)



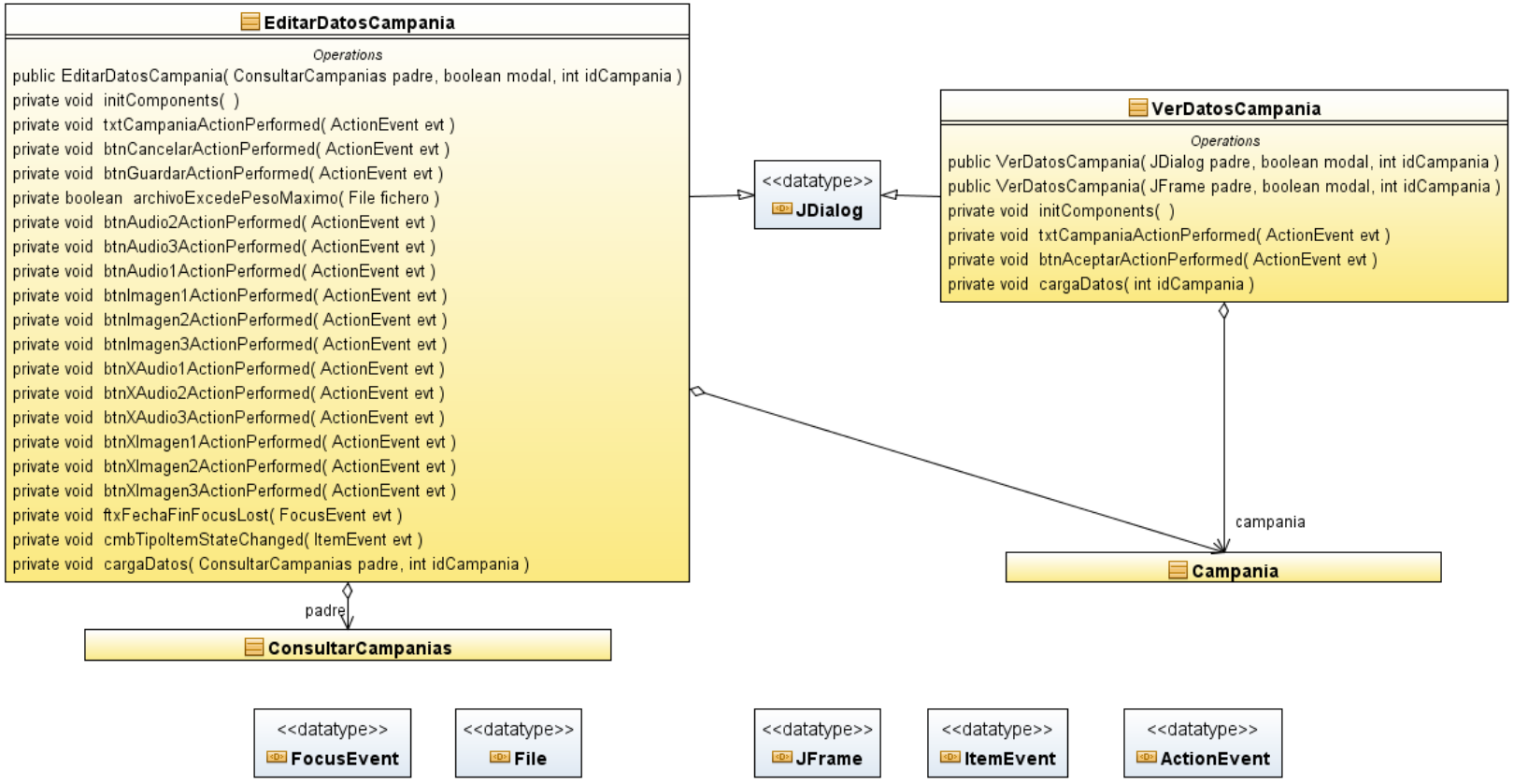

<<datatype>>
PlainDocument
<<datatype>>
AttributeSet
<<datatype>>
Exception
FormateadorDocumento
Attributes
private int longitudMaxima
private boolean conLongitudMaxima
private int modo
private Vector vecLetras
private Vector vecNumeros
Operations
public FormateadorDocumento(int longitudMaxima, int modo)
public void setLongitudMaxima(int longitudMaxima)
public void setLongitudMaxima(boolean conLongitudMaxima)
public void insertString(int inicio, String strCadena, AttributeSet astAtributos)
ErrorAplicativo
Attributes
private String clase
private String metodo
private String tecnico
Operations
public ErrorAplicativo(String clase, String metodo, String message, String tecnico)
public String getError( )
public String getErrorTecnico( )
public String getClase( )
public void setClase(String clase)
public String getMetodo( )
public void setMetodo(String metodo)
public String getTecnico( )
public void setTecnico(String tecnico)
FormateadorFecha
Attributes
private SimpleDateFormat DATE_JAVA = new SimpleDateFormat("dd/MM/yyyy")
private SimpleDateFormat DATETIME_JAVA = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss")
private SimpleDateFormat HOUR_JAVA = new SimpleDateFormat("HH")
private SimpleDateFormat MINUTE_JAVA = new SimpleDateFormat("mm")
private SimpleDateFormat DATE_DB = new SimpleDateFormat("yyyy-MM-dd")
private SimpleDateFormat DATETIME_DB = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")
private Logger logger = Logger.getLogger(FormateadorFecha.class)
Operations
public String aJavaDateString(Date fecha)
public String aJavaDateTimeString(Date fecha)
public String aJavaHourString(Date fecha)
public String aJavaMinuteString(Date fecha)
public String aDbDateString(Date fecha)
public String aDbDateTimeString(Date fecha)
public Date aJavaDateDate(String fecha)
public Date aJavaDateTimeDate(String fecha)
Conexion
Attributes
private Connection conexion = null
private String bd = "bluecatch"
private String usuario = "root"
private String clave = "mysql"
private String url = "jdbc:mysql://localhost/" + bd
private Logger logger = Logger.getLogger(Conexion.class)
Operations
public Connection getConexion( )
protected void inicializaConexion( )
<<datatype>>
Connection
<<datatype>>
String
<<datatype>>
Date

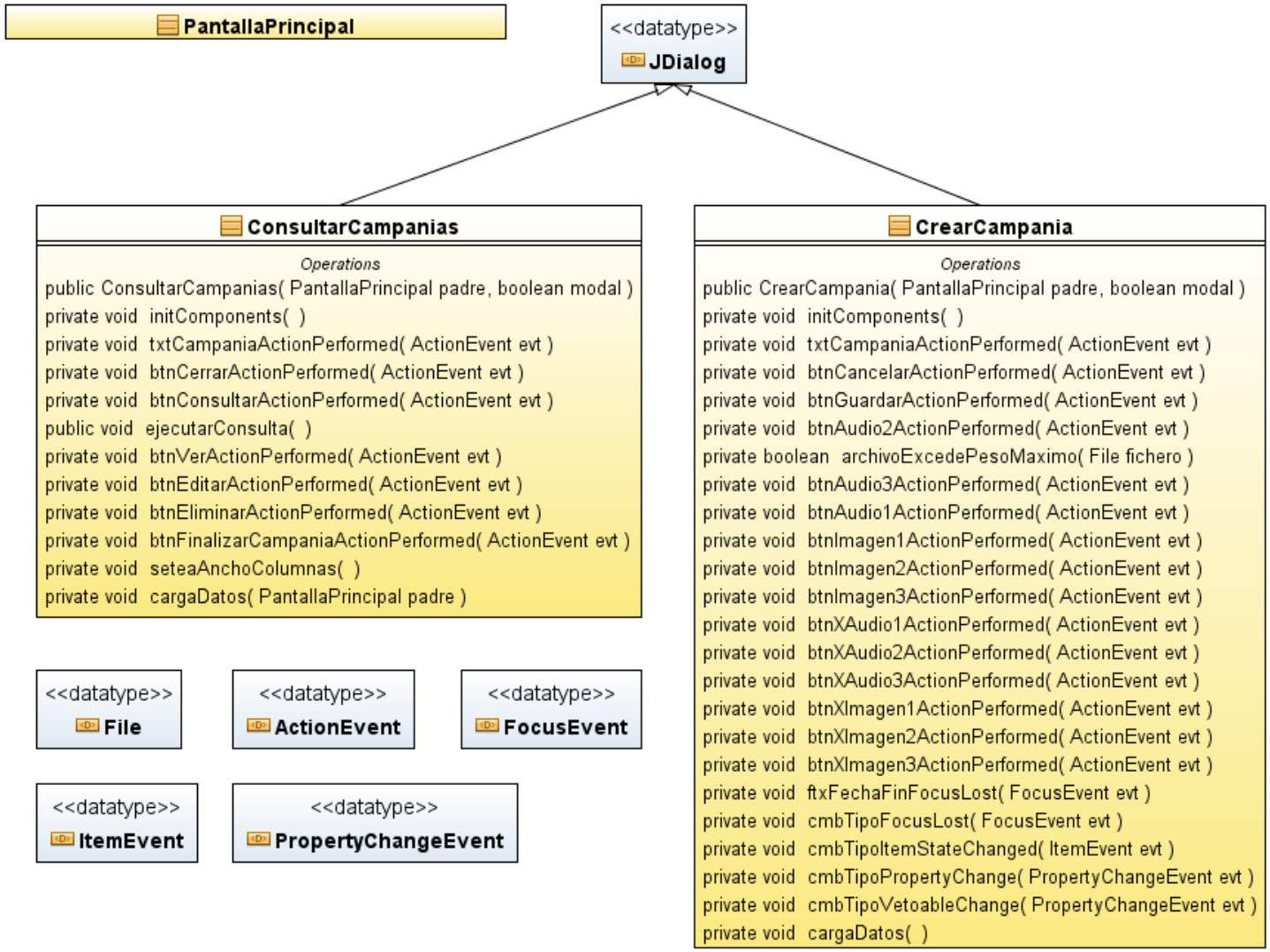
3.2.7 Clases para objetos de interfaz gráfica de usuario correspondientes al menú de sistema (com.jabluecatch.gui.sistema)



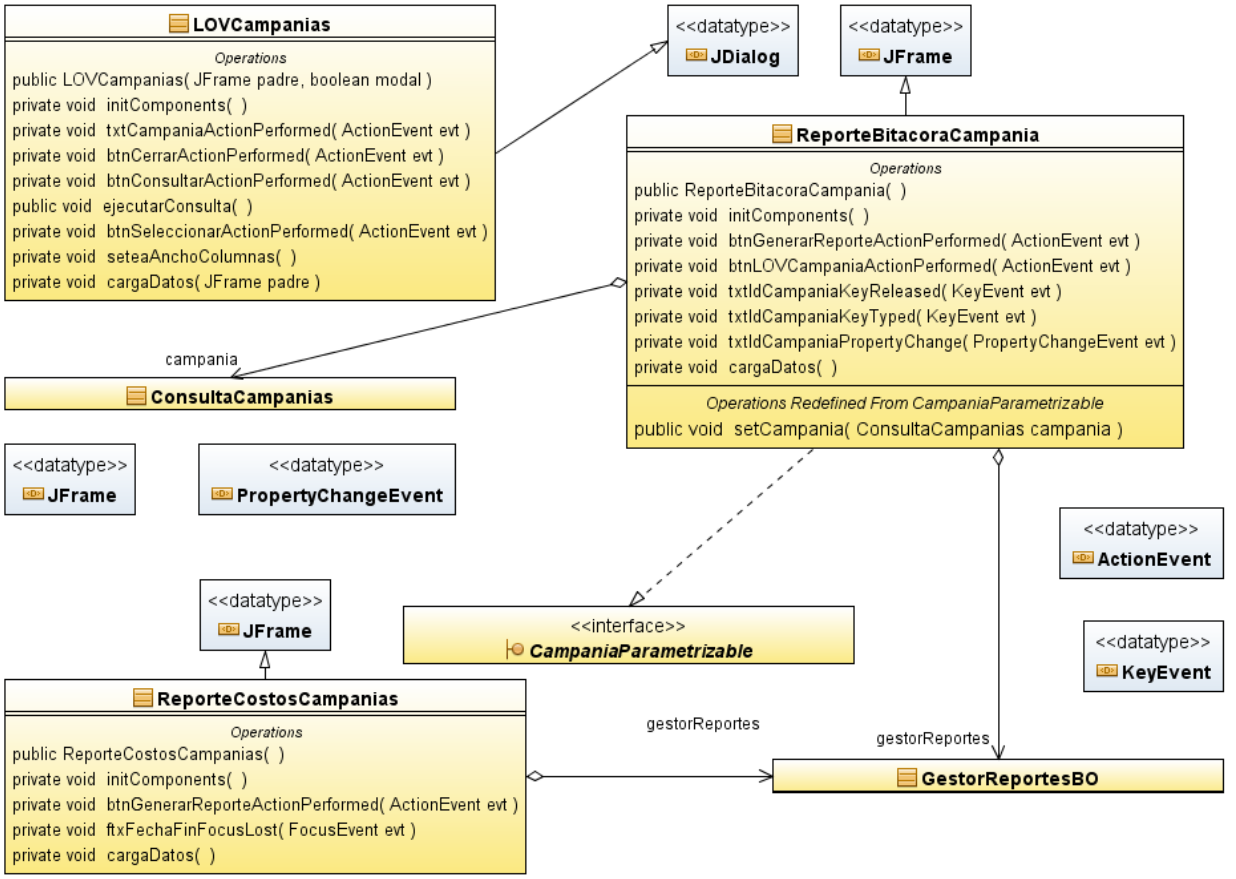

EditarDatosUsuario
Operations
public EditarDatosUsuario(ConsultarUsuarios padre, boolean modal, String nombreUsuario)
private void initComponents( )
private void txtUsuarioActionPerformed(ActionEvent evt)
private void btnCancelarActionPerformed(ActionEvent evt)
private void btnGuardarActionPerformed(ActionEvent evt)
private void cargaDatos(ConsultarUsuarios padre, String nombreUsuario)
<<datatype>>
JDialog
usuario
Usuario
padre
ConsultarUsuarios
<<datatype>>
ActionEvent
VerDatosUsuario
Operations
public VerDatosUsuario(JDialog padre, boolean modal, String nombreUsuario)
private void initComponents( )
private void txtUsuarioActionPerformed(ActionEvent evt)
private void btnAceptarActionPerformed(ActionEvent evt)
private void cargaDatos(String nombreUsuario)
usuario
Usuario
<<datatype>>
String
ParametrosGenerales
Operations
public ParametrosGenerales(PantallaPrincipal padre, boolean modal)
private void initComponents( )
private void btnCerrarActionPerformed(ActionEvent evt)
private void btnGuardarActionPerformed(ActionEvent evt)
private void btnRutaReportesActionPerformed(ActionEvent evt)
private void txtRutaReportesActionPerformed(ActionEvent evt)
private void cargaDatos( )
tiempoBloqueo
ParametroGeneral
rutaReportes

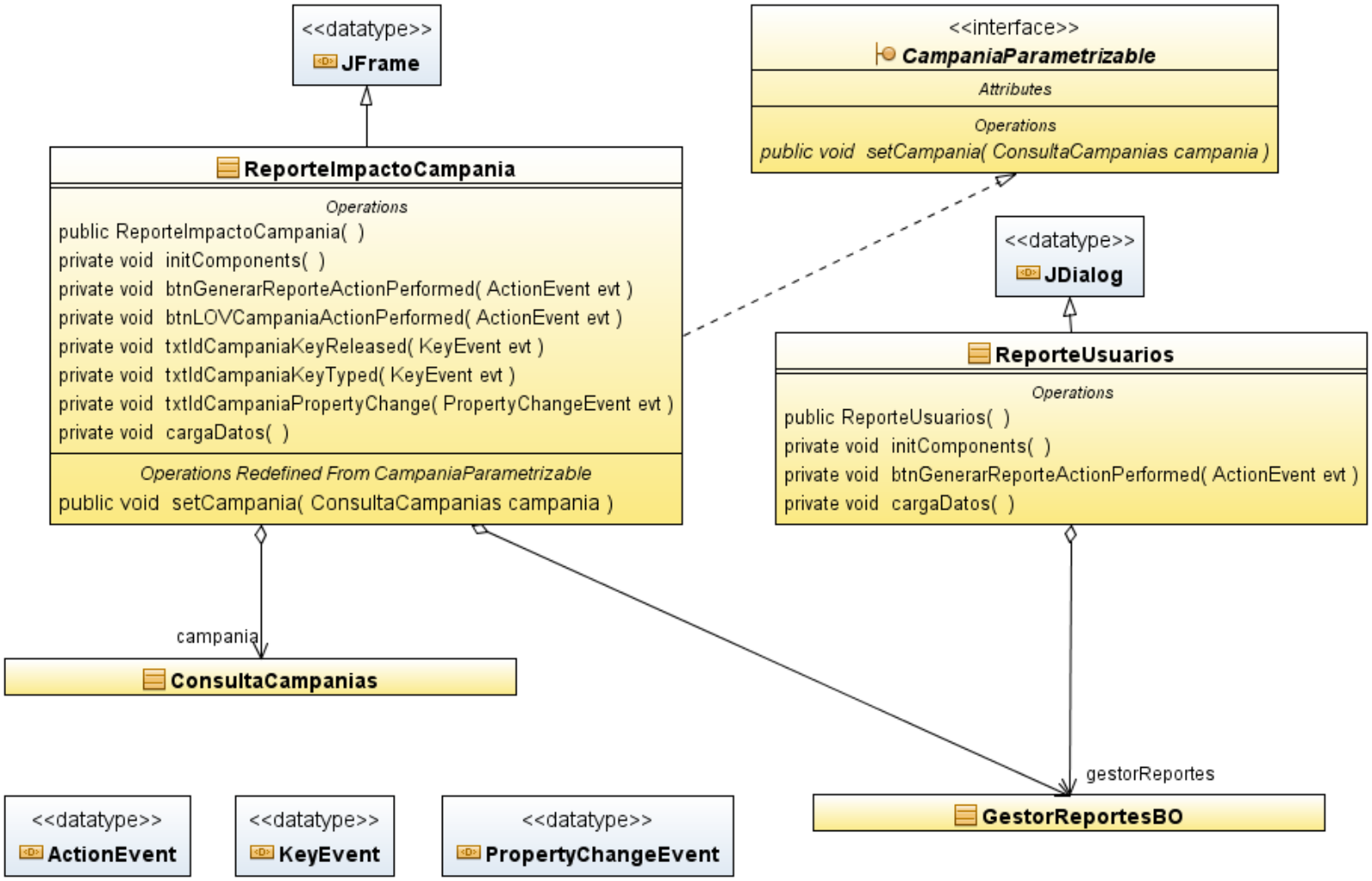
3.2.8 Clases para objetos de interfaz gráfica de usuario correspondientes al menú de Campaña (com.jabluecatch.gui.campania)


EditarDatosCampania
Operations
public EditarDatosCampania(ConsultarCampanias padre, boolean modal, int idCampania)
private void initComponents( )
private void txtCampaniaActionPerformed(ActionEvent evt)
private void btnCancelarActionPerformed(ActionEvent evt)
private void btnGuardarActionPerformed(ActionEvent evt)
private boolean archivoExcedePesoMaximo(File fichero)
private void btnAudio2ActionPerformed(ActionEvent evt)
private void btnAudio3ActionPerformed(ActionEvent evt)
private void btnAudio1ActionPerformed(ActionEvent evt)
private void btnImagen1ActionPerformed(ActionEvent evt)
private void btnImagen2ActionPerformed(ActionEvent evt)
private void btnImagen3ActionPerformed(ActionEvent evt)
private void btnXAudio1ActionPerformed(ActionEvent evt)
private void btnXAudio2ActionPerformed(ActionEvent evt)
private void btnXAudio3ActionPerformed(ActionEvent evt)
private void btnXImagen1ActionPerformed(ActionEvent evt)
private void btnXImagen2ActionPerformed(ActionEvent evt)
private void btnXImagen3ActionPerformed(ActionEvent evt)
private void ftxFechaFinFocusLost(FocusEvent evt)
private void cmbTipoItemStateChanged(ItemEvent evt)
private void cargaDatos(ConsultarCampanias padre, int idCampania)
<<datatype>>
JDialog
VerDatosCampania
Operations
public VerDatosCampania(JDialog padre, boolean modal, int idCampania)
public VerDatosCampania(JFrame padre, boolean modal, int idCampania)
private void initComponents( )
private void txtCampaniaActionPerformed(ActionEvent evt)
private void btnAceptarActionPerformed(ActionEvent evt)
private void cargaDatos(int idCampania)
campania
Campania
padre
ConsultarCampanias
<<datatype>>
FocusEvent
<<datatype>>
File
<<datatype>>
JFrame
<<datatype>>
ItemEvent
<<datatype>>
ActionEvent

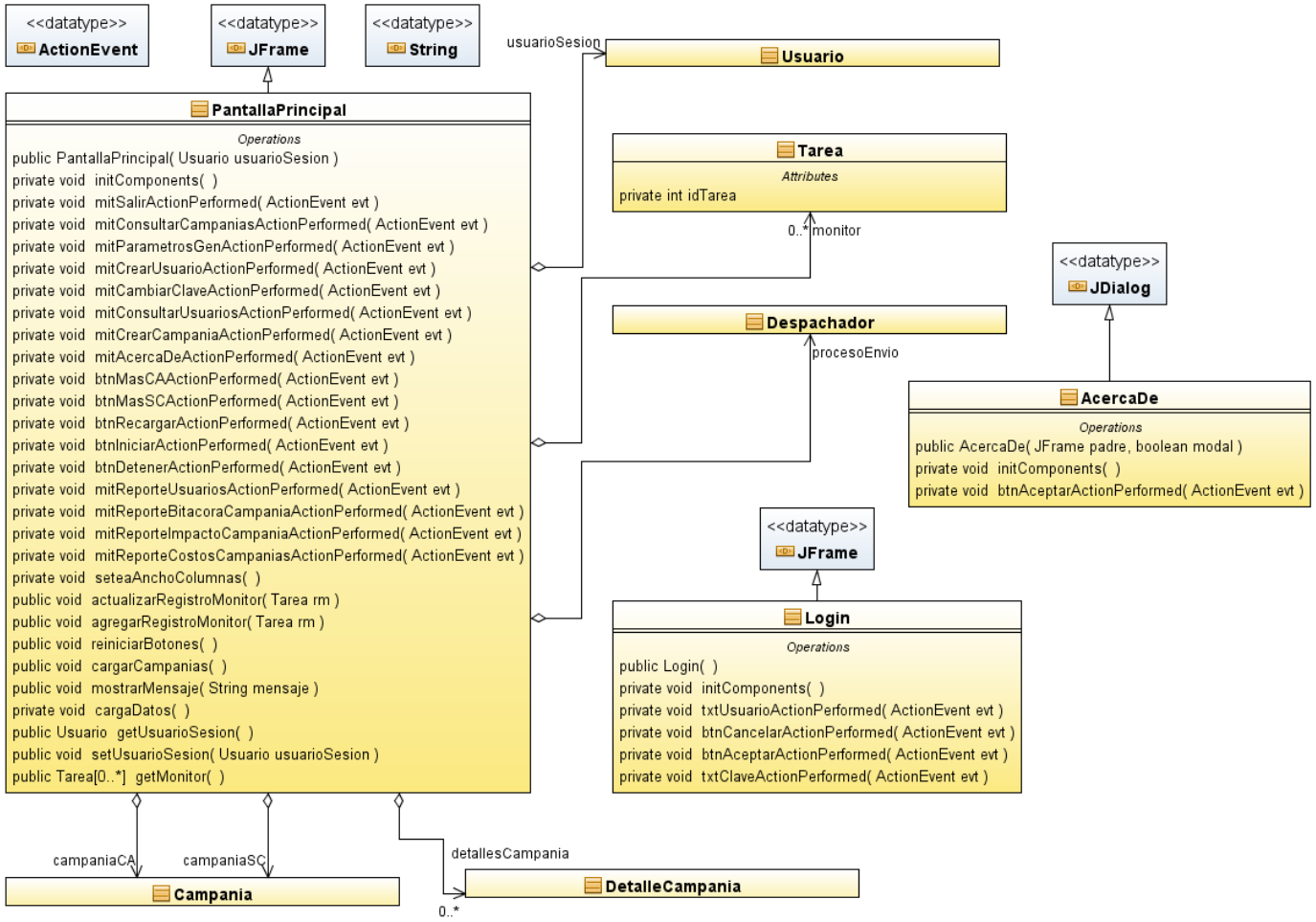

PantallaPrincipal
<<datatype>>
JDialog
ConsultarCampanias
Operations
public ConsultarCampanias(PantallaPrincipal padre, boolean modal)
private void initComponents( )
private void txtCampaniaActionPerformed(ActionEvent evt)
private void btnCerrarActionPerformed(ActionEvent evt)
private void btnConsultarActionPerformed(ActionEvent evt)
public void ejecutarConsulta( )
private void btnVerActionPerformed(ActionEvent evt)
private void btnEditarActionPerformed(ActionEvent evt)
private void btnEliminarActionPerformed(ActionEvent evt)
private void btnFinalizarCampaniaActionPerformed(ActionEvent evt)
private void seteaAnchoColumnas( )
private void cargaDatos(PantallaPrincipal padre)
CrearCampania
Operations
public CrearCampania(PantallaPrincipal padre, boolean modal)
private void initComponents( )
private void txtCampaniaActionPerformed(ActionEvent evt)
private void btnCancelarActionPerformed(ActionEvent evt)
private void btnGuardarActionPerformed(ActionEvent evt)
private void btnAudio2ActionPerformed(ActionEvent evt)
private boolean archivoExcedePesoMaximo(File fichero)
private void btnAudio3ActionPerformed(ActionEvent evt)
private void btnAudio1ActionPerformed(ActionEvent evt)
private void btnImagen1ActionPerformed(ActionEvent evt)
private void btnImagen2ActionPerformed(ActionEvent evt)
private void btnImagen3ActionPerformed(ActionEvent evt)
private void btnXAudio1ActionPerformed(ActionEvent evt)
private void btnXAudio2ActionPerformed(ActionEvent evt)
private void btnXAudio3ActionPerformed(ActionEvent evt)
private void btnXImagen1ActionPerformed(ActionEvent evt)
private void btnXImagen2ActionPerformed(ActionEvent evt)
private void btnXImagen3ActionPerformed(ActionEvent evt)
private void ftxFechaFinFocusLost(FocusEvent evt)
private void cmbTipoFocusLost(FocusEvent evt)
private void cmbTipoItemStateChanged(ItemEvent evt)
private void cmbTipoPropertyChange(PropertyChangeEvent evt)
private void cmbTipoVetoableChange(PropertyChangeEvent evt)
private void cargaDatos( )
<<datatype>>
File
<<datatype>>
ActionEvent
<<datatype>>
FocusEvent
<<datatype>>
ItemEvent
<<datatype>>
PropertyChangeEvent

3.2.9 Clases para objetos de interfaz gráfica de usuario correspondientes al menú de Reportes (com.jabluecatch.gui.reportes)




<<datatype>>
JFrame
<<interface>>
CampaniaParametrizable
Attributes
Operations
public void setCampania(ConsultaCampanias campania)
ReporteImpactoCampania
Operations
public ReporteImpactoCampania( )
private void initComponents( )
private void btnGenerarReporteActionPerformed(ActionEvent evt)
private void btnLOVCampaniaActionPerformed(ActionEvent evt)
private void txtIdCampaniaKeyReleased(KeyEvent evt)
private void txtIdCampaniaKeyTyped(KeyEvent evt)
private void txtIdCampaniaPropertyChange(PropertyChangeEvent evt)
private void cargaDatos( )
Operations Redefined From CampaniaParametrizable
public void setCampania(ConsultaCampanias campania)
<<datatype>>
JDialog
ReporteUsuarios
Operations
public ReporteUsuarios( )
private void initComponents( )
private void btnGenerarReporteActionPerformed(ActionEvent evt)
private void cargaDatos( )
campania
ConsultaCampanias
gestorReportes
GestorReportesBO
<<datatype>>
ActionEvent
<<datatype>>
KeyEvent
<<datatype>>
PropertyChangeEvent

3.2.10 Clases para objetos de interfaz gráfica de usuario generales (com.jabluecatch.gui)



4 CÓDIGO FUENTE DE LA APLICACIÓN

4.1 CLASES E INTERFACES JAVA

4.1.1 Paquete com.jabluecatch.entidades

4.1.1.1 BitacoraCampania

```
package com.jabluecatch.entidades;

import java.util.Date;

/**
 *POJO para mapeo de la entidad BITACORA_CAMPANIAS
 * @author Fausto Almeida
 */
public class BitacoraCampania {

    private int idBitacoraCampania;
    private int idDetalleCampania;
    private int idUsuario;
    private int idDispositivoDestino;
    private Date fechaRegistro;
    private String evento;
    private String observacion;
    private String estado;

    public BitacoraCampania() {
    }

    public BitacoraCampania(int idDetalleCampania, int idUsuario, int
idDispositivoDestino, Date fechaRegistro, String evento, String observacion, String
estado) {
        this.idDetalleCampania = idDetalleCampania;
        this.idUsuario = idUsuario;
        this.idDispositivoDestino = idDispositivoDestino;
        this.fechaRegistro = fechaRegistro;
        this.evento = evento;
        this.observacion = observacion;
        this.estado = estado;
    }

    public BitacoraCampania(int idBitacoraCampania, int idDetalleCampania, int
idUsuario, int idDispositivoDestino, Date fechaRegistro, String evento, String
observacion, String estado) {
        this.idBitacoraCampania = idBitacoraCampania;
        this.idDetalleCampania = idDetalleCampania;
        this.idUsuario = idUsuario;
        this.idDispositivoDestino = idDispositivoDestino;
        this.fechaRegistro = fechaRegistro;
        this.evento = evento;
        this.observacion = observacion;
```

```
    this.estado = estado;
}

/**
 * @return the idBitacoraCampania
 */
public int getIdBitacoraCampania() {
    return idBitacoraCampania;
}

/**
 * @param idBitacoraCampania the idBitacoraCampania to set
 */
public void setIdBitacoraCampania(int idBitacoraCampania) {
    this.idBitacoraCampania = idBitacoraCampania;
}

/**
 * @return the idDetalleCampania
 */
public int getIdDetalleCampania() {
    return idDetalleCampania;
}

/**
 * @param idDetalleCampania the idDetalleCampania to set
 */
public void setIdDetalleCampania(int idDetalleCampania) {
    this.idDetalleCampania = idDetalleCampania;
}

/**
 * @return the idUsuario
 */
public int getIdUsuario() {
    return idUsuario;
}

/**
 * @param idUsuario the idUsuario to set
 */
public void setIdUsuario(int idUsuario) {
    this.idUsuario = idUsuario;
}

/**
 * @return the idDispositivoDestino
 */
public int getIdDispositivoDestino() {
    return idDispositivoDestino;
}

/**
 * @param idDispositivoDestino the idDispositivoDestino to set
 */
public void setIdDispositivoDestino(int idDispositivoDestino) {
    this.idDispositivoDestino = idDispositivoDestino;
}
```

```
    /**
     * @return the fecha_registro
     */
    public Date getFechaRegistro() {
        return fechaRegistro;
    }

    /**
     * @param fecha_registro the fecha_registro to set
     */
    public void setFechaRegistro(Date fechaRegistro) {
        this.fechaRegistro = fechaRegistro;
    }

    /**
     * @return the evento
     */
    public String getEvento() {
        return evento;
    }

    /**
     * @param evento the evento to set
     */
    public void setEvento(String evento) {
        this.evento = evento;
    }

    /**
     * @return the observacion
     */
    public String getObservacion() {
        return observacion;
    }

    /**
     * @param observacion the observacion to set
     */
    public void setObservacion(String observacion) {
        this.observacion = observacion;
    }

    /**
     * @return the estado
     */
    public String getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.1.2 Campania

```
package com.jabluecatch.entidades;

import java.util.Date;

/**
 *POJO para mapeo de la entidad CAMPANIAS
 * @author Fausto Almeida
 */
public class Campania {

    private int idCampania;
    private String campania;
    private String descripcion;
    private String tipo;
    private Float costoRefMensaje;
    private Date fechaInicio;
    private Date fechaFin;
    private String fase;
    private String estado;

    public Campania() {
    }

    public Campania(String campania, String descripcion, String tipo, Float
costoRefMensaje, Date fechaInicio, Date fechaFin, String fase, String estado) {
        this.campania = campania;
        this.descripcion = descripcion;
        this.tipo = tipo;
        this.costoRefMensaje = costoRefMensaje;
        this.fechaInicio = fechaInicio;
        this.fechaFin = fechaFin;
        this.fase = fase;
        this.estado = estado;
    }

    public Campania(int idCampania, String campania, String descripcion, String
tipo, Float costoRefMensaje, Date fechaInicio, Date fechaFin, String fase, String
estado) {
        this.idCampania = idCampania;
        this.campania = campania;
        this.descripcion = descripcion;
        this.tipo = tipo;
        this.costoRefMensaje = costoRefMensaje;
        this.fechaInicio = fechaInicio;
        this.fechaFin = fechaFin;
        this.fase = fase;
        this.estado = estado;
    }

    /**
     * @return the idCampania
     */
    public int getIdCampania() {
        return idCampania;
```

```
}

/**
 * @param idCampania the idCampania to set
 */
public void setIdCampania(int idCampania) {
    this.idCampania = idCampania;
}

/**
 * @return the campania
 */
public String getCampania() {
    return campania;
}

/**
 * @param campania the campania to set
 */
public void setCampania(String campania) {
    this.campania = campania;
}

/**
 * @return the descripcion
 */
public String getDescripcion() {
    return descripcion;
}

/**
 * @param descripcion the descripcion to set
 */
public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
}

/**
 * @return the tipo
 */
public String getTipo() {
    return tipo;
}

/**
 * @param tipo the tipo to set
 */
public void setTipo(String tipo) {
    this.tipo = tipo;
}

/**
 * @return the costoRefMensaje
 */
public Float getCostoRefMensaje() {
    return costoRefMensaje;
}
```

```
/**
 * @param costoRefMensaje the costoRefMensaje to set
 */
public void setCostoRefMensaje(Float costoRefMensaje) {
    this.costoRefMensaje = costoRefMensaje;
}

/**
 * @return the fechaInicio
 */
public Date getFechaInicio() {
    return fechaInicio;
}

/**
 * @param fechaInicio the fechaInicio to set
 */
public void setFechaInicio(Date fechaInicio) {
    this.fechaInicio = fechaInicio;
}

/**
 * @return the fechaFin
 */
public Date getFechaFin() {
    return fechaFin;
}

/**
 * @param fechaFin the fechaFin to set
 */
public void setFechaFin(Date fechaFin) {
    this.fechaFin = fechaFin;
}

/**
 * @return the fase
 */
public String getFase() {
    return fase;
}

/**
 * @param fase the fase to set
 */
public void setFase(String fase) {
    this.fase = fase;
}

/**
 * @return the estado
 */
public String getEstado() {
    return estado;
}

/**
 * @param estado the estado to set
```

```
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.1.3 DetalleCampania

```
package com.jabluecatch.entidades;

/**
 *POJO para mapeo de la entidad DETALLE_CAMPANIAS
 * @author Fausto Almeida
 */
public class DetalleCampania {

    private int idDetalleCampania;
    private int idCampania;
    private String ruta;
    private String tipo;
    private String estado;

    public DetalleCampania() {
    }

    public DetalleCampania(String ruta, String tipo, String estado) {
        this.ruta = ruta;
        this.tipo = tipo;
        this.estado = estado;
    }

    public DetalleCampania(int idCampania, String ruta, String tipo, String estado)
{
        this.idCampania = idCampania;
        this.ruta = ruta;
        this.tipo = tipo;
        this.estado = estado;
    }

    public DetalleCampania(int idDetalleCampania, int idCampania, String ruta,
String tipo, String estado) {
        this.idDetalleCampania = idDetalleCampania;
        this.idCampania = idCampania;
        this.ruta = ruta;
        this.tipo = tipo;
        this.estado = estado;
    }

    /**
     * @return the idDetalleCampania
     */
    public int getIdDetalleCampania() {
        return idDetalleCampania;
    }
```

```
/**
 * @param idDetalleCampania the idDetalleCampania to set
 */
public void setIdDetalleCampania(int idDetalleCampania) {
    this.idDetalleCampania = idDetalleCampania;
}

/**
 * @return the idCampania
 */
public int getIdCampania() {
    return idCampania;
}

/**
 * @param idCampania the idCampania to set
 */
public void setIdCampania(int idCampania) {
    this.idCampania = idCampania;
}

/**
 * @return the ruta
 */
public String getRuta() {
    return ruta;
}

/**
 * @param ruta the ruta to set
 */
public void setRuta(String ruta) {
    this.ruta = ruta;
}

/**
 * @return the tipo
 */
public String getTipo() {
    return tipo;
}

/**
 * @param tipo the tipo to set
 */
public void setTipo(String tipo) {
    this.tipo = tipo;
}

/**
 * @return the estado
 */
public String getEstado() {
    return estado;
}

/**
 * @param estado the estado to set
```

```
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.1.4 Dispositivo

```
import java.util.Date;

/**
 *POJO para mapeo de la entidad DISPOSITIVOS
 * @author Fausto Almeida
 */
public class Dispositivo {

    private int idDispositivo;
    private String direccionBt;
    private String nombreBt;
    private String estadoBloqueado;
    private Date fechaUltimoBloqueo;
    private String estado;

    public Dispositivo() {
    }

    public Dispositivo(String direccionBt, String estadoBloqueado, String estado) {
        this.direccionBt = direccionBt;
        this.estadoBloqueado = estadoBloqueado;
        this.estado = estado;
    }

    public Dispositivo(String direccionBt, String nombreBt, String estadoBloqueado,
Date fechaUltimoBloqueo, String estado) {
        this.direccionBt = direccionBt;
        this.nombreBt = nombreBt;
        this.estadoBloqueado = estadoBloqueado;
        this.fechaUltimoBloqueo = fechaUltimoBloqueo;
        this.estado = estado;
    }

    public Dispositivo(int idDispositivo, String direccionBt, String nombreBt,
String estadoBloqueado, Date fechaUltimoBloqueo, String estado) {
        this.idDispositivo = idDispositivo;
        this.direccionBt = direccionBt;
        this.nombreBt = nombreBt;
        this.estadoBloqueado = estadoBloqueado;
        this.fechaUltimoBloqueo = fechaUltimoBloqueo;
        this.estado = estado;
    }

    /**
     * @return the idDispositivo
     */
    public int getIdDispositivo() {
```

```
    return idDispositivo;
}

/**
 * @param idDispositivo the idDispositivo to set
 */
public void setIdDispositivo(int idDispositivo) {
    this.idDispositivo = idDispositivo;
}

/**
 * @return the direccionBt
 */
public String getDireccionBt() {
    return direccionBt;
}

/**
 * @param direccionBt the direccionBt to set
 */
public void setDireccionBt(String direccionBt) {
    this.direccionBt = direccionBt;
}

/**
 * @return the nombreBt
 */
public String getNombreBt() {
    return nombreBt;
}

/**
 * @param nombreBt the nombreBt to set
 */
public void setNombreBt(String nombreBt) {
    this.nombreBt = nombreBt;
}

/**
 * @return the estadoBloqueado
 */
public String getEstadoBloqueado() {
    return estadoBloqueado;
}

/**
 * @param estadoBloqueado the estadoBloqueado to set
 */
public void setEstadoBloqueado(String estadoBloqueado) {
    this.estadoBloqueado = estadoBloqueado;
}

/**
 * @return the fechaUltimoBloqueo
 */
public Date getFechaUltimoBloqueo() {
    return fechaUltimoBloqueo;
}
```

```
    /**
     * @param fechaUltimoBloqueo the fechaUltimoBloqueo to set
     */
    public void setFechaUltimoBloqueo(Date fechaUltimoBloqueo) {
        this.fechaUltimoBloqueo = fechaUltimoBloqueo;
    }

    /**
     * @return the estado
     */
    public String getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.1.5 ParametroGeneral

```
package com.jabluecatch.entidades;

/**
 *POJO para mapeo de la entidad PARAMETROS_GENERALES
 * @author Fausto Almeida
 */
public class ParametroGeneral {

    private int idParametroGeneral;
    private String parametro;
    private String descripcion;
    private String valor;
    private String tipo;
    private String estado;

    public ParametroGeneral() {
    }

    public ParametroGeneral(String parametro, String descripcion, String valor,
String tipo, String estado) {
        this.parametro = parametro;
        this.descripcion = descripcion;
        this.valor = valor;
        this.tipo = tipo;
        this.estado = estado;
    }

    public ParametroGeneral(int idParametroGeneral, String parametro, String
descripcion, String valor, String tipo, String estado) {
        this.idParametroGeneral = idParametroGeneral;
```

```
    this.parametro = parametro;
    this.descripcion = descripcion;
    this.valor = valor;
    this.tipo = tipo;
    this.estado = estado;
}

/**
 * @return the idParametroGeneral
 */
public int getIdParametroGeneral() {
    return idParametroGeneral;
}

/**
 * @param idParametroGeneral the idParametroGeneral to set
 */
public void setIdParametroGeneral(int idParametroGeneral) {
    this.idParametroGeneral = idParametroGeneral;
}

/**
 * @return the parametro
 */
public String getParametro() {
    return parametro;
}

/**
 * @param parametro the parametro to set
 */
public void setParametro(String parametro) {
    this.parametro = parametro;
}

/**
 * @return the descripcion
 */
public String getDescripcion() {
    return descripcion;
}

/**
 * @param descripcion the descripcion to set
 */
public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
}

/**
 * @return the valor
 */
public String getValor() {
    return valor;
}

/**
 * @param valor the valor to set
```

```
     */
    public void setValor(String valor) {
        this.valor = valor;
    }

    /**
     * @return the tipo
     */
    public String getTipo() {
        return tipo;
    }

    /**
     * @param tipo the tipo to set
     */
    public void setTipo(String tipo) {
        this.tipo = tipo;
    }

    /**
     * @return the estado
     */
    public String getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.1.6 Rol

```
package com.jabluecatch.entidades;

/**
 *POJO para mapeo de la entidad ROLES
 * @author Fausto Almeida
 */
public class Rol {

    private int idRol;
    private String rol;
    private String estado;

    public Rol() {
    }

    public Rol(String rol, String estado) {
        this.rol = rol;
        this.estado = estado;
    }
```

```
    public Rol(int idRol, String rol, String estado) {
        this.idRol = idRol;
        this.rol = rol;
        this.estado = estado;
    }

    /**
     * @return the idRol
     */
    public int getIdRol() {
        return idRol;
    }

    /**
     * @param idRol the idRol to set
     */
    public void setIdRol(int idRol) {
        this.idRol = idRol;
    }

    /**
     * @return the rol
     */
    public String getRol() {
        return rol;
    }

    /**
     * @param rol the rol to set
     */
    public void setRol(String rol) {
        this.rol = rol;
    }

    /**
     * @return the estado
     */
    public String getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.1.7 Usuario

```
package com.jabluecatch.entidades;

/**
```

```
 *POJO para mapeo de la entidad USUARIOS
 * @author Fausto Almeida
 */
public class Usuario {

    private int idUsuario;
    private int idRol;
    private String usuario;
    private String clave;
    private String nombres;
    private String apellidos;
    private String identificacion;
    private String estado;

    public Usuario() {
    }

    public Usuario(int id_rol, String usuario, String clave, String nombres, String
apellidos, String identificacion, String estado) {
        this.idRol = id_rol;
        this.usuario = usuario;
        this.clave = clave;
        this.nombres = nombres;
        this.apellidos = apellidos;
        this.identificacion = identificacion;
        this.estado = estado;
    }

    public Usuario(int id_usuario, int id_rol, String usuario, String clave, String
nombres, String apellidos, String identificacion, String estado) {
        this.idUsuario = id_usuario;
        this.idRol = id_rol;
        this.usuario = usuario;
        this.clave = clave;
        this.nombres = nombres;
        this.apellidos = apellidos;
        this.identificacion = identificacion;
        this.estado = estado;
    }

    /**
     * @return the id_usuario
     */
    public int getIdUsuario() {
        return idUsuario;
    }

    /**
     * @param id_usuario the id_usuario to set
     */
    public void setIdUsuario(int idUsuario) {
        this.idUsuario = idUsuario;
    }

    /**
     * @return the id_rol
     */
    public int getIdRol() {
```

```
    return idRol;
}

/**
 * @param id_rol the id_rol to set
 */
public void setIdRol(int idRol) {
    this.idRol = idRol;
}

/**
 * @return the usuario
 */
public String getUsuario() {
    return usuario;
}

/**
 * @param usuario the usuario to set
 */
public void setUsuario(String usuario) {
    this.usuario = usuario;
}

/**
 * @return the clave
 */
public String getClave() {
    return clave;
}

/**
 * @param clave the clave to set
 */
public void setClave(String clave) {
    this.clave = clave;
}

/**
 * @return the nombres
 */
public String getNombres() {
    return nombres;
}

/**
 * @param nombres the nombres to set
 */
public void setNombres(String nombres) {
    this.nombres = nombres;
}

/**
 * @return the apellidos
 */
public String getApellidos() {
    return apellidos;
}
```

```
    /**
     * @param apellidos the apellidos to set
     */
    public void setApellidos(String apellidos) {
        this.apellidos = apellidos;
    }

    /**
     * @return the identificacion
     */
    public String getIdentificacion() {
        return identificacion;
    }

    /**
     * @param identificacion the identificacion to set
     */
    public void setIdentificacion(String identificacion) {
        this.identificacion = identificacion;
    }

    /**
     * @return the estado
     */
    public String getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(String estado) {
        this.estado = estado;
    }
}
```

4.1.2 Paquete com.jabluecatch.dto

4.1.2.1 ConsultaCampanias

```
package com.jabluecatch.dto;

/**
 * POJO para representación de una consulta de campañas para la ventana de
 * consulta y para la lista de valores (LOV)
 * @author Fausto Almeida
 */
public class ConsultaCampanias {
```

```
    private int idCampania;
    private String campania;
    private String tipo;
    private String fechaInicio;
    private String fechaFin;
    private String fase;

    public ConsultaCampanias() {
    }

    public ConsultaCampanias(String campania, String tipo, String fechaInicio,
String fechaFin, String fase) {
        this.campania = campania;
        this.tipo = tipo;
        this.fechaInicio = fechaInicio;
        this.fechaFin = fechaFin;
        this.fase = fase;
    }

    public ConsultaCampanias(int idCampania, String campania, String tipo, String
fechaInicio, String fechaFin, String fase) {
        this.idCampania = idCampania;
        this.campania = campania;
        this.tipo = tipo;
        this.fechaInicio = fechaInicio;
        this.fechaFin = fechaFin;
        this.fase = fase;
    }

    /**
     * @return the idCampania
     */
    public int getIdCampania() {
        return idCampania;
    }

    /**
     * @param idCampania the idCampania to set
     */
    public void setIdCampania(int idCampania) {
        this.idCampania = idCampania;
    }

    /**
     * @return the campania
     */
    public String getCampania() {
        return campania;
    }

    /**
     * @param campania the campania to set
     */
    public void setCampania(String campania) {
        this.campania = campania;
    }

    /**
```

```
     * @return the tipo
     */
    public String getTipo() {
        return tipo;
    }

    /**
     * @param tipo the tipo to set
     */
    public void setTipo(String tipo) {
        this.tipo = tipo;
    }

    /**
     * @return the fechaInicio
     */
    public String getFechaInicio() {
        return fechaInicio;
    }

    /**
     * @param fechaInicio the fechaInicio to set
     */
    public void setFechaInicio(String fechaInicio) {
        this.fechaInicio = fechaInicio;
    }

    /**
     * @return the fechaFin
     */
    public String getFechaFin() {
        return fechaFin;
    }

    /**
     * @param fechaFin the fechaFin to set
     */
    public void setFechaFin(String fechaFin) {
        this.fechaFin = fechaFin;
    }

    /**
     * @return the fase
     */
    public String getFase() {
        return fase;
    }

    /**
     * @param fase the fase to set
     */
    public void setFase(String fase) {
        this.fase = fase;
    }
}
```

4.1.2.2 ConsultaUsuarios

```
package com.jabluecatch.dto;

/**
 * POJO para representación de una consulta de usuarios para la ventana de
 * consulta de usuarios
 * @author Fausto Almeida
 */
public class ConsultaUsuarios {

    private int idUsuario;
    private String usuario;
    private String rol;
    private String nombres;
    private String apellidos;

    public ConsultaUsuarios() {
    }

    public ConsultaUsuarios(String usuario, String rol, String nombres, String
apellidos) {
        this.usuario = usuario;
        this.rol = rol;
        this.nombres = nombres;
        this.apellidos = apellidos;
    }

    public ConsultaUsuarios(int idUsuario, String usuario, String rol, String
nombres, String apellidos) {
        this.idUsuario = idUsuario;
        this.usuario = usuario;
        this.rol = rol;
        this.nombres = nombres;
        this.apellidos = apellidos;
    }

    /**
     * @return the idUsuario
     */
    public int getIdUsuario() {
        return idUsuario;
    }

    /**
     * @param idUsuario the idUsuario to set
     */
    public void setIdUsuario(int idUsuario) {
        this.idUsuario = idUsuario;
    }

    /**
     * @return the usuario
     */
    public String getUsuario() {
        return usuario;
```

```
    }

    /**
     * @param usuario the usuario to set
     */
    public void setUsuario(String usuario) {
        this.usuario = usuario;
    }

    /**
     * @return the rol
     */
    public String getRol() {
        return rol;
    }

    /**
     * @param rol the rol to set
     */
    public void setRol(String rol) {
        this.rol = rol;
    }

    /**
     * @return the nombres
     */
    public String getNombres() {
        return nombres;
    }

    /**
     * @param nombres the nombres to set
     */
    public void setNombres(String nombres) {
        this.nombres = nombres;
    }

    /**
     * @return the apellidos
     */
    public String getApellidos() {
        return apellidos;
    }

    /**
     * @param apellidos the apellidos to set
     */
    public void setApellidos(String apellidos) {
        this.apellidos = apellidos;
    }
}
```

4.1.2.3 Tarea

```
package com.jabluecatch.dto;
```

```
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.entidades.Dispositivo;
import javax.bluetooth.RemoteDevice;

/**
 * POJO para representación de un registro de tarea en el monitor del sistema
 * @author Fausto Almeida
 */
public class Tarea {

    //private BitacoraCampania bitacora;
    private int idTarea;
    private RemoteDevice servidor;
    private Dispositivo receptor;
    private String urlServidor;
    private DetalleCampania archivoAEnviar;
    private String estado;
    private String porcentajeCompletado;

    public Tarea() {
    }

    public Tarea(int idTarea, RemoteDevice servidor, Dispositivo receptor,
DetalleCampania archivoAEnviar, String estado, String porcentajeCompletado) {
        this.idTarea = idTarea;
        this.servidor = servidor;
        this.receptor = receptor;
        this.archivoAEnviar = archivoAEnviar;
        this.estado = estado;
        this.porcentajeCompletado = porcentajeCompletado;
    }

    /**
     * @return the idTarea
     */
    public int getIdTarea() {
        return idTarea;
    }

    /**
     * @return the servidor
     */
    public RemoteDevice getServidor() {
        return servidor;
    }

    /**
     * @param servidor the servidor to set
     */
    public void setServidor(RemoteDevice servidor) {
        this.servidor = servidor;
    }

    /**
     * @return the receptor
     */
    public Dispositivo getReceptor() {
```

```
    return receptor;
}

/**
 * @param receptor the receptor to set
 */
public void setReceptor(Dispositivo receptor) {
    this.receptor = receptor;
}

/**
 * @return the urlServidor
 */
public String getUrlServidor() {
    return urlServidor;
}

/**
 * @param urlServidor the urlServidor to set
 */
public void setUrlServidor(String urlServidor) {
    this.urlServidor = urlServidor;
}

/**
 * @return the archivoAEnviar
 */
public DetalleCampania getArchivoAEnviar() {
    return archivoAEnviar;
}

/**
 * @param archivoAEnviar the archivoAEnviar to set
 */
public void setArchivoAEnviar(DetalleCampania archivoAEnviar) {
    this.archivoAEnviar = archivoAEnviar;
}

/**
 * @return the estado
 */
public String getEstado() {
    return estado;
}

/**
 * @param estado the estado to set
 */
public void setEstado(String estado) {
    this.estado = estado;
}

/**
 * @return the porcentajeEnviado
 */
public String getPorcentajeCompletado() {
    return porcentajeCompletado;
}
```

```
    /**
     * @param porcentajeEnviado the porcentajeEnviado to set
     */
    public void setPorcentajeCompletado(String porcentajeCompletado) {
        this.porcentajeCompletado = porcentajeCompletado;
    }
}
```

4.1.3 Paquete com.jabluecatch.dao

4.1.3.1 BitacoraCampaniaDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.entidades.BitacoraCampania;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
import java.util.Date;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad BITACORA_CAMPANIAS
 * @author Fas
 */
public class BitacoraCampaniaDAO {

    private static Logger logger = Logger.getLogger(BitacoraCampaniaDAO.class);

    public static void insertar(BitacoraCampania bc) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("INSERT INTO
bitacora_campanias (id_detalle_campania, id_usuario, id_dispositivo_destino,
fecha_registro, evento, observacion, estado) "
                    + "VALUES (?, ?, ?, ?, ?, ?, ?)");

            pst.setInt(1, bc.getIdDetalleCampania());
            pst.setInt(2, bc.getIdUsuario());
            pst.setInt(3, bc.getIdDispositivoDestino());
            pst.setTimestamp(4, new Timestamp(bc.getFechaRegistro().getTime()));
            pst.setString(5, bc.getEvento());
            pst.setString(6, bc.getObservacion());
```

```
            pst.setString(7, bc.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("BitacoraCampaniaDAO", "insertar", "Error al
ejecutar el ingreso", sqle.getMessage());
        }
    }

    public static void actualizar(BitacoraCampania bc) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE bitacora_campanias
SET id_detalle_campania = ? "
                    + ", id_usuario = ? "
                    + ", id_dispositivo_destino = ? "
                    + ", fecha_registro = ? "
                    + ", evento = ? "
                    + ", observacion = ? "
                    + ", estado = ? WHERE id_bitacora_campania = ? ");

            pst.setInt(1, bc.getIdDetalleCampania());
            pst.setInt(2, bc.getIdUsuario());
            pst.setInt(3, bc.getIdDispositivoDestino());
            pst.setTimestamp(4, new Timestamp(bc.getFechaRegistro().getTime()));
            pst.setString(5, bc.getEvento());
            pst.setString(6, bc.getObservacion());
            pst.setString(7, bc.getEstado());
            pst.setInt(8, bc.getIdBitacoraCampania());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("BitacoraCampaniaDAO", "actualizar", "Error
al ejecutar la actualizacion", sqle.getMessage());
        }
    }

    public static BitacoraCampania buscarPorPK(int idBitacoraCampania) throws
ErrorAplicativo {

        BitacoraCampania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT
id_bitacora_campania, "
                    + "id_detalle_campania, "
                    + "id_usuario, "
                    + "id_dispositivo_destino, "
```

```
                    + "fecha_registro, "
                    + "evento, "
                    + "observacion, "
                    + "estado "
                    + "FROM bitacora_campanias WHERE id_bitacora_campania = ? "
                    + "AND estado = 'A'");

            pst.setInt(1, idBitacoraCampania);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new BitacoraCampania(rs.getInt(1),
                        rs.getInt(2),
                        rs.getInt(3),
                        rs.getInt(4),
                        rs.getTimestamp(5),
                        rs.getString(6),
                        rs.getString(7),
                        rs.getString(8));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPK - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("BitacoraCampaniaDAO", "buscarPorPK", "Error
al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static BitacoraCampania buscarIdDispositivoDestinoYFechaRegistro(int
idDispositivoDestino, Date fechaRegistro) throws ErrorAplicativo {

        BitacoraCampania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT
id_bitacora_campania, "
                    + "id_detalle_campania, "
                    + "id_usuario, "
                    + "id_dispositivo_destino, "
                    + "fecha_registro, "
                    + "evento, "
                    + "observacion, "
                    + "estado "
                    + "FROM bitacora_campanias WHERE id_dispositivo_destino = ? "
                    + "AND fecha_registro = ? "
                    + "AND estado = 'A'");

            pst.setInt(1, idDispositivoDestino);
            pst.setTimestamp(2, new Timestamp(fechaRegistro.getTime()));
```

```
            rs = pst.executeQuery();

            while (rs.next()) {
                result = new BitacoraCampania(rs.getInt(1),
                        rs.getInt(2),
                        rs.getInt(3),
                        rs.getInt(4),
                        rs.getTimestamp(5),
                        rs.getString(6),
                        rs.getString(7),
                        rs.getString(8));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPK - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("BitacoraCampaniaDAO", "buscarPorPK", "Error
al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.2 CampaniaDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.entidades.Campania;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
import java.util.Date;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad CAMPANIAS
 * @author Fas
 */
public class CampaniaDAO {

    private static Logger logger = Logger.getLogger(CampaniaDAO.class);

    public static void insertar(Campania ca) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
```

```
            PreparedStatement pst = c.prepareStatement("INSERT INTO campanias
(campania, descripcion, tipo, costo_ref_mensaje, fecha_inicio, fecha_fin, fase,
estado) "
                    + "VALUES (?, ?, ?, ?, ?, ?, ?, ?)");

            pst.setString(1, ca.getCampania());
            pst.setString(2, ca.getDescripcion());
            pst.setString(3, ca.getTipo());
            pst.setFloat(4, ca.getCostoRefMensaje());
            pst.setTimestamp(5, new Timestamp(ca.getFechaInicio().getTime()));
            pst.setTimestamp(6, (ca.getTipo().equals("O") ? new
Timestamp(ca.getFechaFin().getTime()) : null));
            pst.setString(7, ca.getFase());
            pst.setString(8, ca.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "insertar", "Error al ejecutar
el ingreso", sqle.getMessage());
        }
    }

    public static void actualizar(Campania ca) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE campanias SET
campania = ? "
                    + ", descripcion = ? "
                    + ", tipo = ? "
                    + ", costo_ref_mensaje = ? "
                    + ", fecha_inicio = ? "
                    + ", fecha_fin = ? "
                    + ", fase = ? "
                    + ", estado = ? WHERE id_campania = ? ");

            pst.setString(1, ca.getCampania());
            pst.setString(2, ca.getDescripcion());
            pst.setString(3, ca.getTipo());
            pst.setFloat(4, ca.getCostoRefMensaje());
            pst.setTimestamp(5, new Timestamp(ca.getFechaInicio().getTime()));
            pst.setTimestamp(6, (ca.getFechaFin() != null ? new
Timestamp(ca.getFechaFin().getTime()) : null));
            pst.setString(7, ca.getFase());
            pst.setString(8, ca.getEstado());
            pst.setInt(9, ca.getIdCampania());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "actualizar", "Error al
ejecutar la actualizacion", sqle.getMessage());
```

```
        }
    }

    public static Campania buscarPorPK(int idCampania) throws ErrorAplicativo {

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE id_campania = ? "
                    + "AND estado = 'A'");

            pst.setInt(1, idCampania);

            rs = pst.executeQuery();
            while (rs.next()) {
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPK - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "buscarPorPK", "Error al
ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Campania buscarPorFechaReferencial(Date fechaReferencial) throws
ErrorAplicativo {

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
```

```
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE ? "
                    + "BETWEEN fecha_inicio AND IFNULL(fecha_fin,NOW()) "
                    + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fechaReferencial.getTime()));

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorFechaReferencial - Error al ejecutar la consulta
- " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "buscarPorFechaReferencial",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Campania buscarPorFechaInicio(Date fechaInicio) throws
ErrorAplicativo {

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE fecha_inicio = ? "
```

```
                + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fechaInicio.getTime()));

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorFechaReferencial - Error al ejecutar la consulta
- " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "buscarPorFechaReferencial",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Campania buscarPorIdCampaniaYFechaReferencial(int idCampania,
Date fechaReferencial) throws ErrorAplicativo {

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE ? "
                    + "BETWEEN fecha_inicio AND fecha_fin "
                    + "AND id_campania <> ? "
                    + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fechaReferencial.getTime()));
            pst.setInt(2, idCampania);

            rs = pst.executeQuery();

            while (rs.next()) {
```

```
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorIdCampaniaYFechaReferencial - Error al ejecutar
la consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO",
"buscarPorIdCampaniaYFechaReferencial", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Campania buscarPorFechaInicioMasCercana() throws ErrorAplicativo
{

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE fecha_inicio > NOW() "
                    + "AND estado = 'A' "
                    + "ORDER BY fecha_inicio DESC "
                    + "LIMIT 1");

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
```

```
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorFechaInicioMasCercana - Error al ejecutar la
consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO",
"buscarPorFechaInicioMasCercana", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Campania buscarPorFechaInicioMasCercana(Date fecha) throws
ErrorAplicativo {

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE fecha_inicio > ? "
                    + "AND estado = 'A' "
                    + "ORDER BY fecha_inicio DESC "
                    + "LIMIT 1");

            pst.setTimestamp(1, new Timestamp(fecha.getTime()));

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorFechaInicioMasCercana - Error al ejecutar la
consulta - " + sqle.getMessage());
```

```
            throw new ErrorAplicativo("CampaniaDAO",
"buscarPorFechaInicioMasCercana", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Campania buscarPorOtraFechaInicioMasCercana(int idCampania, Date
fecha) throws ErrorAplicativo {

        Campania result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE fecha_inicio > ? "
                    + "AND id_campania <> ? "
                    + "AND estado = 'A' "
                    + "ORDER BY fecha_inicio DESC "
                    + "LIMIT 1");

            pst.setTimestamp(1, new Timestamp(fecha.getTime()));
            pst.setInt(2, idCampania);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorFechaInicioMasCercana - Error al ejecutar la
consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO",
"buscarPorFechaInicioMasCercana", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
```

```
        }
        return result;
    }

    public static Vector buscarCampaniasRodeadas(Date fechaInicio, Date fechaFin)
throws ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE (fecha_inicio BETWEEN ? AND ? "
                    + "OR fecha_fin BETWEEN ? AND ? ) "
                    + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fechaInicio.getTime()));
            pst.setTimestamp(2, new Timestamp(fechaFin.getTime()));
            pst.setTimestamp(3, new Timestamp(fechaInicio.getTime()));
            pst.setTimestamp(4, new Timestamp(fechaFin.getTime()));

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarCampaniasRodeadas - Error al ejecutar la consulta -
" + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "buscarCampaniasRodeadas",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarOtrasCampaniasRodeadas(int idCampania, Date
fechaInicio, Date fechaFin) throws ErrorAplicativo {
```

```
        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE (fecha_inicio BETWEEN ? AND ? "
                    + "OR fecha_fin BETWEEN ? AND ? ) "
                    + "AND id_campania <> ? "
                    + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fechaInicio.getTime()));
            pst.setTimestamp(2, new Timestamp(fechaFin.getTime()));
            pst.setTimestamp(3, new Timestamp(fechaInicio.getTime()));
            pst.setTimestamp(4, new Timestamp(fechaFin.getTime()));
            pst.setInt(5, idCampania);

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarCampaniasRodeadas - Error al ejecutar la consulta -
" + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "buscarCampaniasRodeadas",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarCampaniasPermanentesIniciadas(Date fecha) throws
ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;
```

```
        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE ? "
                    + "BETWEEN fecha_inicio AND ? "
                    + "AND tipo = 'P' "
                    + "AND fase <> 'F' "
                    + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fecha.getTime()));
            pst.setTimestamp(2, new Timestamp(fecha.getTime()));

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarCampaniasPermanentesIniciadas - Error al ejecutar la
consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO",
"buscarCampaniasPermanentesIniciadas", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarOtrasCampaniasPermanentesIniciadas(int idCampania,
Date fecha) throws ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
```

```
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
                    + "fecha_fin, "
                    + "fase, "
                    + "estado "
                    + "FROM campanias WHERE ? "
                    + "BETWEEN fecha_inicio AND ? "
                    + "AND tipo = 'P' "
                    + "AND fase <> 'F' "
                    + "AND id_campania <> ? "
                    + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fecha.getTime()));
            pst.setTimestamp(2, new Timestamp(fecha.getTime()));
            pst.setInt(3, idCampania);

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarCampaniasPermanentesIniciadas - Error al ejecutar la
consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO",
"buscarCampaniasPermanentesIniciadas", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarCampaniasFinalizadas(Date fecha) throws
ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_campania, "
                    + "campania, "
                    + "descripcion, "
                    + "tipo, "
                    + "costo_ref_mensaje, "
                    + "fecha_inicio, "
```

```
                + "fecha_fin, "
                + "fase, "
                + "estado "
                + "FROM campanias WHERE fecha_fin <= ? "
                + "AND estado = 'A'");

            pst.setTimestamp(1, new Timestamp(fecha.getTime()));

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new Campania(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getFloat(5),
                        rs.getTimestamp(6),
                        (rs.getString(4).equals("O") ? rs.getTimestamp(7) : null),
                        rs.getString(8),
                        rs.getString(9)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarCampaniasFinalizadas - Error al ejecutar la consulta
- " + sqle.getMessage());
            throw new ErrorAplicativo("CampaniaDAO", "buscarCampaniasFinalizadas",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.3 ConsultaCampaniasDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.dto.ConsultaCampanias;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad de transferencia
 * ConsultaCampanias
 * @author Fas
 */
public class ConsultaCampaniasDAO {
```

```
    private static Logger logger = Logger.getLogger(ConsultaCampaniasDAO.class);

    public static Vector buscarPorPatronCampania(String patronCampania) throws
ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT c.id_campania, "
                    + "c.campania, "
                    + "c.tipo, "
                    + "DATE_FORMAT(c.fecha_inicio,'%d/%m/%Y %H:%i:%S'), "
                    + "DATE_FORMAT(c.fecha_fin,'%d/%m/%Y %H:%i:%S'), "
                    + "c.fase "
                    + "FROM campanias c WHERE c.campania LIKE ? "
                    + "AND c.estado = 'A' "
                    + "ORDER BY fecha_inicio");

            pst.setString(1, "%" + patronCampania + "%");

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new ConsultaCampanias(rs.getInt(1),
                        rs.getString(2),
                        (rs.getString(3).equals("O") ? "Operativa" : "Automática"),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPatronCampania - Error al ejecutar la consulta -
" + sqle.getMessage());
            throw new ErrorAplicativo("ConsultaCampaniasDAO",
"buscarPorPatronCampania", "Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarPorFechaReferencial(String fechaReferencial) throws
ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {

            PreparedStatement pst = c.prepareStatement("SELECT c.id_campania, "
                    + "c.campania, "
                    + "c.tipo, "
```

```
                    + "DATE_FORMAT(c.fecha_inicio,'%d/%m/%Y %H:%i:%S'), "
                    + "DATE_FORMAT(c.fecha_fin,'%d/%m/%Y %H:%i:%S'), "
                    + "c.fase "
                    + "FROM campanias c WHERE
DATE_FORMAT(c.fecha_inicio,'%d/%m/%Y') LIKE ? "
                    + "OR DATE_FORMAT(c.fecha_fin,'%d/%m/%Y') LIKE ? "
                    + "AND c.estado = 'A' "
                    + "ORDER BY fecha_inicio");

            pst.setString(1, "%" + fechaReferencial + "%");
            pst.setString(2, "%" + fechaReferencial + "%");

            rs = pst.executeQuery();
            while (rs.next()) {
                result.add(new ConsultaCampanias(rs.getInt(1),
                        rs.getString(2),
                        (rs.getString(3).equals("O") ? "Operativa" : "Automática"),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorFechaReferencial - Error al ejecutar la consulta
- " + sqle.getMessage());
            throw new ErrorAplicativo("ConsultaCampaniasDAO",
"buscarPorFechaReferencial", "Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarPorPatronCampaniaYFechaReferencial(String
patronCampania, String fechaReferencial) throws ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT c.id_campania, "
                    + "c.campania, "
                    + "c.tipo, "
                    + "DATE_FORMAT(c.fecha_inicio,'%d/%m/%Y %H:%i:%S'), "
                    + "DATE_FORMAT(c.fecha_fin,'%d/%m/%Y %H:%i:%S'), "
                    + "c.fase "
                    + "FROM campanias c WHERE c.campania LIKE ? "
                    + "AND (DATE_FORMAT(c.fecha_inicio,'%d/%m/%Y') LIKE ? "
                    + "OR DATE_FORMAT(c.fecha_fin,'%d/%m/%Y') LIKE ?) "
                    + "AND c.estado = 'A' "
                    + "ORDER BY fecha_inicio");

            pst.setString(1, "%" + patronCampania + "%");
            pst.setString(2, "%" + fechaReferencial + "%");
            pst.setString(3, "%" + fechaReferencial + "%");
```

```
            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new ConsultaCampanias(rs.getInt(1),
                        rs.getString(2),
                        (rs.getString(3).equals("O") ? "Operativa" : "Automática"),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPatronCampaniaYFechaReferencial - Error al
ejecutar la consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("ConsultaCampaniasDAO",
"buscarPorPatronCampaniaYFechaReferencial", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.4 ConsultaUsuariosDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.dto.ConsultaUsuarios;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad de transferencia
 * ConsultaUsuarios
 * @author Fas
 */
public class ConsultaUsuariosDAO {

    private static Logger logger = Logger.getLogger(ConsultaUsuariosDAO.class);

    public static Vector buscarPorPatronUsuario(String patronUsuario) throws
ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;
```

```
        try {
            PreparedStatement pst = c.prepareStatement("SELECT u.id_usuario, "
                    + "u.usuario, "
                    + "r.rol, "
                    + "u.nombres, "
                    + "u.apellidos "
                    + "FROM usuarios u, roles r WHERE u.id_rol = r.id_rol "
                    + "AND u.usuario LIKE ? "
                    + "AND u.estado = 'A' "
                    + "AND r.estado = 'A'");

            pst.setString(1, "%" + patronUsuario + "%");

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new ConsultaUsuarios(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPatronUsuario - Error al ejecutar la consulta - "
+ sqle.getMessage());
            throw new ErrorAplicativo("ConsultaUsuariosDAO",
"buscarPorPatronUsuario", "Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarPorPatronNombreYApellido(String
patronNombreYApellido) throws ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT u.id_usuario, "
                    + "u.usuario, "
                    + "r.rol, "
                    + "u.nombres, "
                    + "u.apellidos "
                    + "FROM usuarios u, roles r WHERE u.id_rol = r.id_rol "
                    + "AND (u.nombres LIKE ? "
                    + "OR u.apellidos LIKE ?) "
                    + "AND u.estado = 'A' "
                    + "AND r.estado = 'A'");

            pst.setString(1, "%" + patronNombreYApellido + "%");
            pst.setString(2, "%" + patronNombreYApellido + "%");

            rs = pst.executeQuery();
```

```
            while (rs.next()) {
                result.add(new ConsultaUsuarios(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPatronNombreYApellido - Error al ejecutar la
consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("ConsultaUsuariosDAO",
"buscarPorPatronNombreYApellido", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarPorPatronUsuarioYNombreYApellido(String
patronUsuario, String patronNombreYApellido) throws ErrorAplicativo {

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT u.id_usuario, "
                    + "u.usuario, "
                    + "r.rol, "
                    + "u.nombres, "
                    + "u.apellidos "
                    + "FROM usuarios u, roles r WHERE u.id_rol = r.id_rol "
                    + "AND u.usuario LIKE ? "
                    + "AND (u.nombres LIKE ? "
                    + "OR u.apellidos LIKE ?) "
                    + "AND u.estado = 'A' "
                    + "AND r.estado = 'A'");

            pst.setString(1, "%" + patronUsuario + "%");
            pst.setString(2, "%" + patronNombreYApellido + "%");
            pst.setString(3, "%" + patronNombreYApellido + "%");

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new ConsultaUsuarios(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5)));
            }
            rs.close();

        } catch (SQLException sqle) {
```

```
            logger.warn("buscarPorPatronUsuarioYNombreYApellido - Error al ejecutar
la consulta - " + sqle.getMessage());
            throw new ErrorAplicativo("ConsultaUsuariosDAO",
"buscarPorPatronUsuarioYNombreYApellido", "Error al ejecutar la consulta",
sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.5 DetalleCampaniaDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.util.Conexion;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad DETALLE_CAMPANIAS
 * @author Fas
 */
public class DetalleCampaniaDAO {

    private static Logger logger = Logger.getLogger(DetalleCampaniaDAO.class);

    public static void insertar(DetalleCampania dc) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("INSERT INTO
detalle_campanias (id_campania, ruta, tipo, estado) "
                    + "VALUES (?, ?, ?, ?)");

            pst.setInt(1, dc.getIdCampania());
            pst.setString(2, dc.getRuta());
            pst.setString(3, dc.getTipo());
            pst.setString(4, dc.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("DetalleCampaniaDAO", "insertar", "Error al
ejecutar el ingreso", sqle.getMessage());
```

```
        }
    }

    public static void actualizar(DetalleCampania dc) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE detalle_campanias
SET id_campania = ? "
                    + ", ruta = ? "
                    + ", tipo = ? "
                    + ", estado = ? WHERE id_detalle_campania = ? ");

            pst.setInt(1, dc.getIdCampania());
            pst.setString(2, dc.getRuta());
            pst.setString(3, dc.getTipo());
            pst.setString(4, dc.getEstado());
            pst.setInt(5, dc.getIdDetalleCampania());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("DetalleCampaniaDAO", "actualizar", "Error al
ejecutar la actualizacion", sqle.getMessage());
        }
    }

    public static void eliminar(DetalleCampania dc) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("DELETE FROM
detalle_campanias WHERE id_detalle_campania = ? ");

            pst.setInt(1, dc.getIdDetalleCampania());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("eliminar - Error al ejecutar la eliminación - " +
sqle.getMessage());
            throw new ErrorAplicativo("DetalleCampaniaDAO", "eliminar", "Error al
ejecutar la eliminación", sqle.getMessage());
        }
    }

    public static Vector buscarPorIdCampania(int idCampania) throws ErrorAplicativo
{

        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
```

```
            PreparedStatement pst = c.prepareStatement("SELECT id_detalle_campania,
"
                    + "id_campania, "
                    + "ruta, "
                    + "tipo, "
                    + "estado "
                    + "FROM detalle_campanias WHERE id_campania = ? "
                    + "AND estado = 'A'");

            pst.setInt(1, idCampania);

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new DetalleCampania(rs.getInt(1),
                        rs.getInt(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPK - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("DetalleCampaniaDAO", "buscarPorPK", "Error
al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.6 DispositivoDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.entidades.Dispositivo;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad DISPOSITIVOS
 * @author Fas
 */
public class DispositivoDAO {

    private static Logger logger = Logger.getLogger(DispositivoDAO.class);
```

```
    public static void insertar(Dispositivo d) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("INSERT INTO dispositivos
(direccion_bt, nombre_bt, estado_bloqueado, fecha_ultimo_bloqueo, estado) "
                    + "VALUES (?, ?, ?, ?, ?)");

            pst.setString(1, d.getDireccionBt());
            pst.setString(2, d.getNombreBt());
            pst.setString(3, d.getEstadoBloqueado());
            pst.setTimestamp(4, (d.getFechaUltimoBloqueo() != null ? new
Timestamp(d.getFechaUltimoBloqueo().getTime()) : null));
            pst.setString(5, d.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("DispositivoDAO", "insertar", "Error al
ejecutar el ingreso", sqle.getMessage());
        }
    }

    public static void actualizar(Dispositivo d) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE dispositivos SET
direccion_bt = ? "
                    + ", nombre_bt = ? "
                    + ", estado_bloqueado = ? "
                    + ", fecha_ultimo_bloqueo = ? "
                    + ", estado = ? WHERE id_dispositivo = ? ");

            pst.setString(1, d.getDireccionBt());
            pst.setString(2, d.getNombreBt());
            pst.setString(3, d.getEstadoBloqueado());
            pst.setTimestamp(4, (d.getFechaUltimoBloqueo() != null ? new
Timestamp(d.getFechaUltimoBloqueo().getTime()) : null));
            pst.setString(5, d.getEstado());
            pst.setInt(6, d.getIdDispositivo());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("DispositivoDAO", "actualizar", "Error al
ejecutar la actualizacion", sqle.getMessage());
        }
    }
```

```
    public static Dispositivo buscarPorPK(int idDispositivo) throws ErrorAplicativo
{

        Dispositivo result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_dispositivo, "
                    + "direccion_bt, "
                    + "nombre_bt, "
                    + "estado_bloqueado, "
                    + "fecha_ultimo_bloqueo, "
                    + "estado "
                    + "FROM dispositivos WHERE id_dispositivo = ? "
                    + "AND estado = 'A'");

            pst.setInt(1, idDispositivo);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Dispositivo(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getTimestamp(5),
                        rs.getString(6));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPK - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("DispositivoDAO", "buscarPorPK", "Error al
ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Dispositivo buscarPorDireccionBt(String direccionBt) throws
ErrorAplicativo {

        Dispositivo result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_dispositivo, "
                    + "direccion_bt, "
                    + "nombre_bt, "
                    + "estado_bloqueado, "
                    + "fecha_ultimo_bloqueo, "
                    + "estado "
                    + "FROM dispositivos WHERE direccion_bt = ? "
                    + "AND estado = 'A'");
```

```
            pst.setString(1, direccionBt);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Dispositivo(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getTimestamp(5),
                        rs.getString(6));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorDireccionBt - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("DispositivoDAO", "buscarPorDireccionBt",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.7 GestorReportesDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Map;
import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperFillManager;
import net.sf.jasperreports.engine.JasperPrint;
import net.sf.jasperreports.engine.JasperReport;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para el gestor de reportes Jasper
 * @author Fas
 */
public class GestorReportesDAO {

    private static Logger logger = Logger.getLogger(GestorReportesDAO.class);

    public static JasperPrint obtenerImpresion(JasperReport reporte, Map
parametros) throws ErrorAplicativo {
        try {
            return JasperFillManager.fillReport(reporte, parametros,
Conexion.getConexion());
        } catch (JRException jre) {
```

```
            logger.warn("obtenerImpresion - Error al obtener el Reporte - " +
jre.getMessage());
            throw new ErrorAplicativo("GestorReportesDAO", "obtenerImpresion",
"Error al obtener la impresión del reporte", jre.getMessage());
        }
    }
}
```

4.1.3.8 ParametroGeneralDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.util.Conexion;
import com.jabluecatch.entidades.ParametroGeneral;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad PARAMETROS_GENERALES
 * @author Fas
 */
public class ParametroGeneralDAO {

    private static Logger logger = Logger.getLogger(ParametroGeneralDAO.class);

    public static void insertar(ParametroGeneral p) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("INSERT INTO
parametros_generales (parametro, descripcion, valor, tipo, estado) "
                    + "VALUES (?, ?, ?, ?, ?)");

            pst.setString(1, p.getParametro());
            pst.setString(2, p.getDescripcion());
            pst.setString(3, p.getValor());
            pst.setString(4, p.getTipo());
            pst.setString(5, p.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("ParametroGeneralDAO", "insertar", "Error al
ejecutar el ingreso", sqle.getMessage());
        }
    }

    public static void actualizar(ParametroGeneral p) throws ErrorAplicativo {
```

```
        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE parametros_generales
SET parametro = ? "
                    + ", descripcion = ? "
                    + ", valor = ? "
                    + ", tipo = ? "
                    + ", estado = ? WHERE id_parametro_general = ? ");

            pst.setString(1, p.getParametro());
            pst.setString(2, p.getDescripcion());
            pst.setString(3, p.getValor());
            pst.setString(4, p.getTipo());
            pst.setString(5, p.getEstado());
            pst.setInt(6, p.getIdParametroGeneral());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("ParametroGeneralDAO", "actualizar", "Error
al ejecutar la actualizacion", sqle.getMessage());
        }
    }

    public static ParametroGeneral buscarPorParametro(String parametro) throws
ErrorAplicativo {

        ParametroGeneral result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT
id_parametro_general, "
                    + "parametro, "
                    + "descripcion, "
                    + "valor, "
                    + "tipo, "
                    + "estado "
                    + "FROM parametros_generales WHERE parametro = ? "
                    + "AND estado = 'A'");

            pst.setString(1, parametro);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new ParametroGeneral(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6));
            }
```

```
            rs.close();
        } catch (SQLException sqle) {
            logger.warn("buscarPorParametro - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("ParametroGeneralDAO", "buscarPorParametro",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.9 RolDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.util.Conexion;
import com.jabluecatch.entidades.Rol;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad ROLES
 * @author Fas
 */
public class RolDAO {

    private static Logger logger = Logger.getLogger(RolDAO.class);

    public static void insertar(Rol r) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("INSERT INTO roles (rol,
estado) "
                    + "VALUES (?, ?)");

            pst.setInt(1, r.getIdRol());
            pst.setString(2, r.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("RolDAO", "insertar", "Error al ejecutar el
ingreso", sqle.getMessage());
        }
```

```
    }

    public static void actualizar(Rol r) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE roles SET rol = ? "
                    + ", estado = ? "
                    + "' WHERE id_rol = ? ");

            pst.setString(1, r.getRol());
            pst.setString(2, r.getEstado());
            pst.setInt(3, r.getIdRol());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("RolDAO", "actualizar", "Error al ejecutar la
actualizacion", sqle.getMessage());
        }
    }

    public static Rol buscarPorPK(int idRol) throws ErrorAplicativo {

        Rol result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_rol, rol, estado
FROM roles WHERE id_rol = ? AND estado = 'A'");

            pst.setInt(1, idRol);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Rol(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorPK - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("RolDAO", "buscarPorPK", "Error al ejecutar
la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Vector buscarTodos() throws ErrorAplicativo {
```

```
        Vector result = new Vector();
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_rol, rol, estado
FROM roles ORDER BY id_rol AND estado = 'A'");

            rs = pst.executeQuery();

            while (rs.next()) {
                result.add(new Rol(rs.getInt(1),
                        rs.getString(2),
                        rs.getString(3)));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarTodos - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("RolDAO", "buscarTodos", "Error al ejecutar
la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.3.10 UsuarioDAO

```
package com.jabluecatch.dao;

import com.jabluecatch.util.Conexion;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import org.apache.log4j.Logger;

/**
 * Clase con lógica de acceso a datos para la entidad USUARIOS
 * @author Fas
 */
public class UsuarioDAO {

    private static Logger logger = Logger.getLogger(UsuarioDAO.class);

    public static void insertar(Usuario u) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();
```

```
        try {
            PreparedStatement pst = c.prepareStatement("INSERT INTO usuarios
(id_rol, usuario, clave, nombres, apellidos, identificacion, estado) "
                    + "VALUES (?, ?, AES_ENCRYPT(?, SUBSTR(?, 1,4)), ?, ?, ?, ?)");

            pst.setInt(1, u.getIdRol());
            pst.setString(2, u.getUsuario());
            pst.setString(3, u.getClave());
            pst.setString(4, u.getUsuario());
            pst.setString(5, u.getNombres());
            pst.setString(6, u.getApellidos());
            pst.setString(7, u.getIdentificacion());
            pst.setString(8, u.getEstado());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar el ingreso - " +
sqle.getMessage());
            throw new ErrorAplicativo("UsuarioDAO", "insertar", "Error al ejecutar
el ingreso", sqle.getMessage());
        }
    }

    public static void actualizar(Usuario u) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        try {
            PreparedStatement pst = c.prepareStatement("UPDATE usuarios SET id_rol
= ? "
                    + ", usuario = ? "
                    + ", clave = AES_ENCRYPT(?, SUBSTR(?, 1,4))"
                    + ", nombres = ? "
                    + ", apellidos = ? "
                    + ", identificacion = ? "
                    + ", estado = ? WHERE id_usuario = ? ");

            pst.setInt(1, u.getIdRol());
            pst.setString(2, u.getUsuario());
            pst.setString(3, u.getClave());
            pst.setString(4, u.getUsuario());
            pst.setString(5, u.getNombres());
            pst.setString(6, u.getApellidos());
            pst.setString(7, u.getIdentificacion());
            pst.setString(8, u.getEstado());
            pst.setInt(9, u.getIdUsuario());

            pst.executeUpdate();

        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la actualizacion - " +
sqle.getMessage());
            throw new ErrorAplicativo("UsuarioDAO", "actualizar", "Error al
ejecutar la actualizacion", sqle.getMessage());
        }
    }
```

```
    public static Usuario buscarPorUsuario(String usuario) throws ErrorAplicativo {

        Usuario result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_usuario, "
                    + "id_rol, "
                    + "usuario, "
                    + "AES_DECRYPT(clave, SUBSTR(usuario, 1, 4)), "
                    + "nombres, "
                    + "apellidos, "
                    + "identificacion, "
                    + "estado "
                    + "FROM usuarios WHERE usuario = ? "
                    + "AND estado = 'A'");

            pst.setString(1, usuario);

            rs = pst.executeQuery();
            while (rs.next()) {
                result = new Usuario(rs.getInt(1),
                        rs.getInt(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6),
                        rs.getString(7),
                        rs.getString(8));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorUsuario - Error al ejecutar la consulta - " +
sqle.getMessage());
            throw new ErrorAplicativo("UsuarioDAO", "buscarPorUsuario", "Error al
ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Usuario buscarPorUsuarioYClave(String usuario, String clave)
throws ErrorAplicativo {

        Usuario result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_usuario, "
                    + "id_rol, "
                    + "usuario, "
                    + "AES_DECRYPT(clave, SUBSTR(usuario, 1, 4)), "
                    + "nombres, "
                    + "apellidos, "
```

```
                    + "identificacion, "
                    + "estado "
                    + "FROM usuarios WHERE usuario = ? AND
AES_DECRYPT(clave,SUBSTR(usuario,1,4)) = ? "
                    + "AND estado = 'A'");

            pst.setString(1, usuario);
            pst.setString(2, clave);

            rs = pst.executeQuery();

            while (rs.next()) {
                result = new Usuario(rs.getInt(1),
                        rs.getInt(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6),
                        rs.getString(7),
                        rs.getString(8));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorUsuarioYClave - Error al ejecutar la consulta - "
+ sqle.getMessage());
            throw new ErrorAplicativo("UsuarioDAO", "buscarPorUsuarioYClave",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }

    public static Usuario buscarPorIdentificacion(String identificacion) throws
ErrorAplicativo {

        Usuario result = null;
        Connection c = Conexion.getConexion();
        ResultSet rs = null;

        try {
            PreparedStatement pst = c.prepareStatement("SELECT id_usuario, "
                    + "id_rol, "
                    + "usuario, "
                    + "AES_DECRYPT(clave, SUBSTR(usuario, 1, 4)), "
                    + "nombres, "
                    + "apellidos, "
                    + "identificacion, "
                    + "estado "
                    + "FROM usuarios WHERE identificacion = ? "
                    + "AND estado = 'A'");

            pst.setString(1, identificacion);

            rs = pst.executeQuery();

            while (rs.next()) {
```

```
                result = new Usuario(rs.getInt(1),
                        rs.getInt(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6),
                        rs.getString(7),
                        rs.getString(8));
            }
            rs.close();

        } catch (SQLException sqle) {
            logger.warn("buscarPorIdentificacion - Error al ejecutar la consulta -
" + sqle.getMessage());
            throw new ErrorAplicativo("UsuarioDAO", "buscarPorIdentificacion",
"Error al ejecutar la consulta", sqle.getMessage());
        } finally {
            rs = null;
        }
        return result;
    }
}
```

4.1.4 Paquete com.jabluecatch.bo

4.1.4.1 BitacoraCampaniaBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.BitacoraCampaniaDAO;
import com.jabluecatch.entidades.BitacoraCampania;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.SQLException;
import org.apache.log4j.Logger;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
 * para la entidad BITACORA_CAMPANIAS
 * @author Fausto Almeida
 */
public class BitacoraCampaniaBO {

    private static Logger logger = Logger.getLogger(BitacoraCampaniaBO.class);

    public static BitacoraCampania insertar(BitacoraCampania bcNueva) throws
ErrorAplicativo {

        Connection c = Conexion.getConexion();
```

```
        BitacoraCampaniaDAO.insertar(bcNueva);

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("BitacoraCampaniaBO", "insertar", "Error al
ejecutar la transacción", sqle.getMessage());
        }

        return
BitacoraCampaniaDAO.buscarIdDispositivoDestinoYFechaRegistro(bcNueva.getIdDispositi
voDestino(), bcNueva.getFechaRegistro());
    }

    public static void actualizar(BitacoraCampania bcNueva) throws ErrorAplicativo
{

        Connection c = Conexion.getConexion();

        //Verificar que la bitacora exista
        BitacoraCampania bc =
BitacoraCampaniaDAO.buscarPorPK(bcNueva.getIdBitacoraCampania());
        if (bc == null) {
            logger.info("actualizar - Bitácora de la campaña no existe en la base
de datos");
            throw new ErrorAplicativo("BitacoraCampania", "actualizar", "itácora de
la campaña no existe en la base de datos", null);
        }

        BitacoraCampaniaDAO.actualizar(bcNueva);

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("BitacoraCampaniaBO", "actualizar", "Error al
ejecutar la transacción", sqle.getMessage());
        }

    }
}
```

4.1.4.2 CampaniaBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.CampaniaDAO;
import com.jabluecatch.dao.DetalleCampaniaDAO;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.util.Conexion;
```

```
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.Calendar;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
 * para la entidad CAMPANIAS
 * @author Fausto Almeida
 */
public class CampaniaBO {

    private static Logger logger = Logger.getLogger(CampaniaBO.class);

    public static void insertar(Campania cNueva, Vector<DetalleCampania>
detallesCampania) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        //Verificar que no existan campanias durante estas fechas
        Campania ca =
CampaniaDAO.buscarPorFechaReferencial(cNueva.getFechaInicio());
        if (ca != null) {
            logger.info("insertar - Ya existe una campania configurada durante la
fecha de inicio ingresada");
            throw new ErrorAplicativo("CampaniaBO", "insertar", "Ya existe una
campania configurada durante la fecha de inicio ingresada", null);
        }

        //Verificar que no existan campanias permanentes iniciadas para la fecha de
inicio ingresada
        Vector campaniasPermanentesIniciadas =
CampaniaDAO.buscarCampaniasPermanentesIniciadas(cNueva.getFechaInicio());
        if (campaniasPermanentesIniciadas.size() > 0) {
            logger.info("insertar - Existen " +
campaniasPermanentesIniciadas.size() + " campañas permanentes iniciadas previamente
para la fecha de inicio ingresada");
            throw new ErrorAplicativo("CampaniaBO", "insertar", "Existen " +
campaniasPermanentesIniciadas.size() + " campañas permanentes iniciadas previamente
para la fecha de inicio ingresada", null);
        }

        if (cNueva.getTipo().equals("O")) {
            //si la campania es operativa, realizar las validaciones por la
fecha_fin
            ca = CampaniaDAO.buscarPorFechaReferencial(cNueva.getFechaFin());
            if (ca != null) {
                logger.info("insertar - Ya existe una campania configurada durante
la fecha de finalización ingresada");
                throw new ErrorAplicativo("CampaniaBO", "insertar", "Ya existe una
campania configurada durante la fecha de finalización ingresada", null);
            }
            //Verificar q no existan campanias rodeadeas entre la fecha inicio y la
fecha fin de la campania q se va a insertar
            Vector campaniasRodeadas =
CampaniaDAO.buscarCampaniasRodeadas(cNueva.getFechaInicio(), cNueva.getFechaFin());
```

```
            if (campaniasRodeadas.size() > 0) {
                logger.info("insertar - Existen " + campaniasRodeadas.size() + "
campañas rodeadas previamente configuradas durante las fechas ingresadas");
                throw new ErrorAplicativo("CampaniaBO", "insertar", "Existen " +
campaniasRodeadas.size() + " campañas rodeadas previamente configuradas durante las
fechas ingresadas", null);
            }

        } else {
            //si la campania es permanente, revisar q no exista otra campania en el
futuro
            ca =
CampaniaDAO.buscarPorFechaInicioMasCercana(cNueva.getFechaInicio());
            if (ca != null) {
                logger.warn("insertar - Ya existe una campania configurada en el
futuro, la campaña que se desea crear no puede ser permanente");
                throw new ErrorAplicativo("CampaniaBO", "insertar", "Ya existe una
campania configurada en el futuro, la campaña que se desea crear no puede ser
permanente", null);
            }
        }

        //Insertar campaña
        CampaniaDAO.insertar(cNueva);
        ca = CampaniaDAO.buscarPorFechaInicio(cNueva.getFechaInicio());

        //Insertar cada detalle
        for (int i = 0; i < detallesCampania.size(); i++) {
            detallesCampania.get(i).setIdCampania(ca.getIdCampania());
            DetalleCampaniaDAO.insertar(detallesCampania.get(i));
        }

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaBO", "insertar", "Error al ejecutar
la transacción", sqle.getMessage());
        }

    }

    public static void actualizar(Campania cNueva) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        //Verificar que la campania exista
        Campania ca = CampaniaDAO.buscarPorPK(cNueva.getIdCampania());
        if (ca == null) {
            logger.info("actualizar - Campaña no existe en la base de datos");
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Campaña no
existe en la base de datos", null);
        }

        //Verificar que no existan campanias durante estas fechas y q no tome en
cuenta la q yo estoy enviando
```

```
        ca =
CampaniaDAO.buscarPorIdCampaniaYFechaReferencial(cNueva.getIdCampania(),
cNueva.getFechaInicio());
        if (ca != null) {
            logger.info("actualizar - Ya existe una campania configurada durante la
fecha de inicio ingresada");
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Ya existe una
campania configurada durante la fecha de inicio ingresada", null);
        }

        //Verificar que no existan campanias permanentes iniciadas para la fecha de
inicio ingresada y que no tome en cuenta la q yo estoy enviando
        Vector campaniasPermanentesIniciadas =
CampaniaDAO.buscarOtrasCampaniasPermanentesIniciadas(cNueva.getIdCampania(),
cNueva.getFechaInicio());
        if (campaniasPermanentesIniciadas.size() > 0) {
            logger.info("actualizar - Existen " +
campaniasPermanentesIniciadas.size() + " campañas permanentes iniciadas previamente
para la fecha de inicio ingresada");
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Existen " +
campaniasPermanentesIniciadas.size() + " campañas permanentes iniciadas previamente
para la fecha de inicio ingresada", null);
        }

        if (cNueva.getTipo().equals("O")) {
            //si la campania es operativa, realizar las validaciones por la
fecha_fin
            ca =
CampaniaDAO.buscarPorIdCampaniaYFechaReferencial(cNueva.getIdCampania(),
cNueva.getFechaFin());
            if (ca != null) {
                logger.info("actualizar - Ya existe una campania configurada
durante la fecha de finalización ingresada");
                throw new ErrorAplicativo("CampaniaBO", "actualizar", "Ya existe
una campania configurada durante la fecha de finalización ingresada", null);
            }
            //Verificar q no existan campanias rodeadeas entre la fecha inicio y la
fecha fin de la campania q se va a actaulizar
            Vector campaniasRodeadas =
CampaniaDAO.buscarOtrasCampaniasRodeadas(cNueva.getIdCampania(),
cNueva.getFechaInicio(), cNueva.getFechaFin());
            if (campaniasRodeadas.size() > 0) {
                logger.info("actualizar - Existen " + campaniasRodeadas.size() + "
campañas rodeadas previamente configuradas durante las fechas ingresadas");
                throw new ErrorAplicativo("CampaniaBO", "actualizar", "Existen " +
campaniasRodeadas.size() + " campañas rodeadas previamente configuradas durante las
fechas ingresadas", null);
            }
        } else {
            //si la campania es permanente, revisar q no exista otra campania en el
futuro
            ca =
CampaniaDAO.buscarPorOtraFechaInicioMasCercana(cNueva.getIdCampania(),
cNueva.getFechaInicio());
            if (ca != null) {
                logger.info("actualizar - Ya existe una campania configurada en el
futuro, la campaña que se desea crear no puede ser permanente");
```

```
                throw new ErrorAplicativo("CampaniaBO", "actualizar", "Ya existe
una campania configurada en el futuro, la campaña que se desea crear no puede ser
permanente", null);
            }
        }

        //Actualizar campaña
        CampaniaDAO.actualizar(cNueva);

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Error al
ejecutar la transacción", sqle.getMessage());
        }

    }

    public static void actualizar(Campania cNueva, Vector<DetalleCampania>
detallesCampania) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        //Verificar que la campania exista
        Campania ca = CampaniaDAO.buscarPorPK(cNueva.getIdCampania());
        if (ca == null) {
            logger.info("actualizar - Campaña no existe en la base de datos");
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Campaña no
existe en la base de datos", null);
        }

        //Verificar que no existan campanias durante estas fechas y q no tome en
cuenta la q yo estoy enviando
        ca =
CampaniaDAO.buscarPorIdCampaniaYFechaReferencial(cNueva.getIdCampania(),
cNueva.getFechaInicio());
        if (ca != null) {
            logger.info("actualizar - Ya existe una campania configurada durante la
fecha de inicio ingresada");
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Ya existe una
campania configurada durante la fecha de inicio ingresada", null);
        }

        //Verificar que no existan campanias permanentes iniciadas para la fecha de
inicio ingresada y que no tome en cuenta la q yo estoy enviando
        Vector campaniasPermanentesIniciadas =
CampaniaDAO.buscarOtrasCampaniasPermanentesIniciadas(cNueva.getIdCampania(),
cNueva.getFechaInicio());
        if (campaniasPermanentesIniciadas.size() > 0) {
            logger.info("actualizar - Existen " +
campaniasPermanentesIniciadas.size() + " campañas permanentes iniciadas previamente
para la fecha de inicio ingresada");
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Existen " +
campaniasPermanentesIniciadas.size() + " campañas permanentes iniciadas previamente
para la fecha de inicio ingresada", null);
        }
```

```
        if (cNueva.getTipo().equals("O")) {
            //si la campania es operativa, realizar las validaciones por la
fecha_fin
            ca =
CampaniaDAO.buscarPorIdCampaniaYFechaReferencial(cNueva.getIdCampania(),
cNueva.getFechaFin());
            if (ca != null) {
                logger.info("actualizar - Ya existe una campania configurada
durante la fecha de finalización ingresada");
                throw new ErrorAplicativo("CampaniaBO", "actualizar", "Ya existe
una campania configurada durante la fecha de finalización ingresada", null);
            }
            //Verificar q no existan campanias rodeadeas entre la fecha inicio y la
fecha fin de la campania q se va a actaulizar
            Vector campaniasRodeadas =
CampaniaDAO.buscarOtrasCampaniasRodeadas(cNueva.getIdCampania(),
cNueva.getFechaInicio(), cNueva.getFechaFin());
            if (campaniasRodeadas.size() > 0) {
                logger.info("actualizar - Existen " + campaniasRodeadas.size() + "
campañas rodeadas previamente configuradas durante las fechas ingresadas");
                throw new ErrorAplicativo("CampaniaBO", "actualizar", "Existen " +
campaniasRodeadas.size() + " campañas rodeadas previamente configuradas durante las
fechas ingresadas", null);
            }
        } else {
            //si la campania es permanente, revisar q no exista otra campania en el
futuro
            ca =
CampaniaDAO.buscarPorOtraFechaInicioMasCercana(cNueva.getIdCampania(),
cNueva.getFechaInicio());
            if (ca != null) {
                logger.warn("actualizar - Ya existe una campania configurada en el
futuro, la campaña que se desea crear no puede ser permanente");
                throw new ErrorAplicativo("CampaniaBO", "actualizar", "Ya existe
una campania configurada en el futuro, la campaña que se desea crear no puede ser
permanente", null);
            }
        }

        //Actualizar campaña
        CampaniaDAO.actualizar(cNueva);

        //Buscar todos sus detalles y eliminarlos
        Vector<DetalleCampania> detallesCampaniaParaEliminar =
DetalleCampaniaDAO.buscarPorIdCampania(cNueva.getIdCampania());
        for (int i = 0; i < detallesCampaniaParaEliminar.size(); i++) {
            DetalleCampaniaDAO.eliminar(detallesCampaniaParaEliminar.get(i));
        }

        //Insertar cada detalle con los valores actualizados
        for (int i = 0; i < detallesCampania.size(); i++) {
            detallesCampania.get(i).setIdCampania(cNueva.getIdCampania());
            DetalleCampaniaDAO.insertar(detallesCampania.get(i));
        }

        try {
            c.commit();
```

```
        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaBO", "actualizar", "Error al
ejecutar la transacción", sqle.getMessage());
        }

    }

    public static void eliminar(Campania cVieja) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        //Verificar que la campania exista
        Campania ca = CampaniaDAO.buscarPorPK(cVieja.getIdCampania());
        if (ca == null) {
            logger.info("eliminar - Campaña no existe en la base de datos");
            throw new ErrorAplicativo("CampaniaBO", "eliminar", "Campaña no existe
en la base de datos", null);
        }

        cVieja.setEstado("I");

        //Actualizar campaña
        CampaniaDAO.actualizar(cVieja);

        //Buscar todos sus detalles e inactivarlos
        Vector<DetalleCampania> detallesCampaniaParaInactivar =
DetalleCampaniaDAO.buscarPorIdCampania(cVieja.getIdCampania());
        for (int i = 0; i < detallesCampaniaParaInactivar.size(); i++) {

detallesCampaniaParaInactivar.get(i).setRuta(detallesCampaniaParaInactivar.get(i).g
etRuta().replace("\\", "\\\\"));
            detallesCampaniaParaInactivar.get(i).setEstado("I");
            DetalleCampaniaDAO.actualizar(detallesCampaniaParaInactivar.get(i));
        }

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("eliminar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaBO", "eliminar", "Error al ejecutar
la transacción", sqle.getMessage());
        }

    }

    public static Campania buscarPorPK(int idCampania) throws ErrorAplicativo {
        return CampaniaDAO.buscarPorPK(idCampania);
    }

    public static void conciliarFases() throws ErrorAplicativo {
        Campania campaniaEnEjecucion =
CampaniaDAO.buscarPorFechaReferencial(Calendar.getInstance().getTime());
        Vector<Campania> campaniasFinalizadas =
CampaniaDAO.buscarCampaniasFinalizadas(Calendar.getInstance().getTime());
        //Actualizar la fase de la campania en ejecucion
```

```
        if ((campaniaEnEjecucion != null) ?
!campaniaEnEjecucion.getFase().equals("E") : false) {
            campaniaEnEjecucion.setFase("E");
            CampaniaBO.actualizar(campaniaEnEjecucion);
        }
        //Actualizar la fase de las campaniaa finalizadas
        for (int i = 0; i < campaniasFinalizadas.size(); i++) {
            if (!campaniasFinalizadas.get(i).getFase().equals("F")) {
                campaniasFinalizadas.get(i).setFase("F");

campaniasFinalizadas.get(i).setFechaFin(Calendar.getInstance().getTime());
                CampaniaBO.actualizar(campaniasFinalizadas.get(i));
            }
        }
    }

    public static Campania obtenerCampaniaEnEjecucion() throws ErrorAplicativo {
        Campania result = null;
        result =
CampaniaDAO.buscarPorFechaReferencial(Calendar.getInstance().getTime());

        result = (result != null ? (!result.getFase().equals("F") ? (result) :
(null)) :(result));

        return result;
    }

    public static Campania obtenerSiguienteCampania() throws ErrorAplicativo {
        return CampaniaDAO.buscarPorFechaInicioMasCercana();
    }

    public static void conciliarFase(int idCampania) throws ErrorAplicativo {
        Connection c = Conexion.getConexion();

        //Verificar que la campania exista
        Campania ca = CampaniaDAO.buscarPorPK(idCampania);
        if (ca == null) {
            logger.info("conciliarFase - Campaña no existe en la base de datos");
            throw new ErrorAplicativo("CampaniaBO", "conciliarFase", "Campaña no
existe en la base de datos", null);
        }

        if (ca.getFechaFin().before(Calendar.getInstance().getTime()) &&
!ca.getFase().equals("F")) {
            ca.setFase("F");
            ca.setFechaFin(Calendar.getInstance().getTime());
            //Actualizar campaña
            CampaniaDAO.actualizar(ca);
        }

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("conciliarFase - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("CampaniaBO", "conciliarFase", "Error al
ejecutar la transacción", sqle.getMessage());
        }
```

```
    }
}
```

4.1.4.3 ConsultaCampaniasBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.ConsultaCampaniasDAO;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;

/**
 * Clase que contiene la lógica de negocios para la entidad de transferencia
 * de consulta de campañas
 * @author Fausto Almeida
 */
public class ConsultaCampaniasBO {

    public static Vector buscarPorPatronCampania(String patronCampania) throws
ErrorAplicativo {
        return ConsultaCampaniasDAO.buscarPorPatronCampania(patronCampania);
    }

    public static Vector buscarPorFechaReferencial(String fechaReferencial) throws
ErrorAplicativo {
        return ConsultaCampaniasDAO.buscarPorFechaReferencial(fechaReferencial);
    }

    public static Vector buscarPorPatronCampaniaYFechaReferencial(String
patronCampania, String fechaReferencial) throws ErrorAplicativo {
        return
ConsultaCampaniasDAO.buscarPorPatronCampaniaYFechaReferencial(patronCampania,
fechaReferencial);
    }
}
```

4.1.4.4 ConsultaUsuariosBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.ConsultaUsuariosDAO;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;

/**
 * Clase que contiene la lógica de negocios para la entidad de transferencia
 * de consulta de usuarios
 * @author Fausto Almeida
 */
public class ConsultaUsuariosBO {
```

```
    public static Vector buscarPorPatronUsuario(String patronUsuario) throws
ErrorAplicativo {
        return ConsultaUsuariosDAO.buscarPorPatronUsuario(patronUsuario);
    }

    public static Vector buscarPorPatronNombreYApellido(String
patronNombreYApellido) throws ErrorAplicativo {
        return
ConsultaUsuariosDAO.buscarPorPatronNombreYApellido(patronNombreYApellido);
    }

    public static Vector buscarPorPatronUsuarioYNombreYApellido(String
patronUsuario, String patronNombreYApellido) throws ErrorAplicativo {
        return
ConsultaUsuariosDAO.buscarPorPatronUsuarioYNombreYApellido(patronUsuario,
patronNombreYApellido);
    }
}
```

4.1.4.5 DetalleCampaniaBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.DetalleCampaniaDAO;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
 * para la entidad DETALLE_CAMPANIAS
 * @author Fausto Almeida
 */
public class DetalleCampaniaBO {

    public static Vector buscarPorIdCampania(int idCampania) throws ErrorAplicativo
{
        return DetalleCampaniaDAO.buscarPorIdCampania(idCampania);
    }
}
```

4.1.4.6 DispositivoBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.DispositivoDAO;
import com.jabluecatch.entidades.Dispositivo;
import com.jabluecatch.entidades.ParametroGeneral;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.Calendar;
```

```
import org.apache.log4j.Logger;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
 * para la entidad DISPOSITIVOS
 * @author Fas
 */
public class DispositivoBO {

    private static Logger logger = Logger.getLogger(DispositivoBO.class);

    public static boolean autorizarParaEnvio(Dispositivo dNuevo) throws
ErrorAplicativo {

        boolean result = true;
        long minutosTranscurridos = 0;
        ParametroGeneral tiempoBloqueo;
        Connection c = Conexion.getConexion();

        //Saber si existe o no el dispositivo
        Dispositivo d =
DispositivoDAO.buscarPorDireccionBt(dNuevo.getDireccionBt());
        if (d == null) {
            DispositivoDAO.insertar(dNuevo);
        } else {

            if (d.getEstadoBloqueado().equals("S")) {

                minutosTranscurridos = (Calendar.getInstance().getTime().getTime()
- d.getFechaUltimoBloqueo().getTime()) / (1000 * 60);
                tiempoBloqueo =
ParametroGeneralBO.buscarPorParametroYEstado("TIEMPO_BLOQUEO","A");

                if (minutosTranscurridos < Long.valueOf(tiempoBloqueo.getValor()))
{
                    result = false;
                } else {
                    d.setEstadoBloqueado("N");
                }
            }
            d.setNombreBt(dNuevo.getNombreBt());
            DispositivoDAO.actualizar(d);
        }

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("autorizarParaEnvio - Error al ejecutar la transacción - "
+ sqle.getMessage());
            throw new ErrorAplicativo("DispositivoBO", "autorizarParaEnvio", "Error
al ejecutar la transacción", sqle.getMessage());
        }

        return result;
    }

    public static void actualizar(Dispositivo dNuevo) throws ErrorAplicativo {
```

```
        Connection c = Conexion.getConexion();

        //Saber si existe o no el dispositivo
        Dispositivo d =
DispositivoDAO.buscarPorDireccionBt(dNuevo.getDireccionBt());
        if (d == null) {
            logger.info("actualizar - Dispositivo no existe en la base de datos");
            throw new ErrorAplicativo("DispositivoBO", "actualizar", "Dispositivo
no existe en la base de datos", null);
        }

        DispositivoDAO.actualizar(dNuevo);

        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("DispositivoBO", "actualizar", "Error al
ejecutar la transacción", sqle.getMessage());
        }

    }

    public static Dispositivo buscarPorDireccionBt(String direccionBt) throws
ErrorAplicativo {
        return DispositivoDAO.buscarPorDireccionBt(direccionBt);
    }
}
```

4.1.4.7 GestorReportesBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.GestorReportesDAO;
import com.jabluecatch.entidades.ParametroGeneral;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Calendar;
import java.util.Map;
import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperExportManager;
import net.sf.jasperreports.engine.JasperPrint;
import net.sf.jasperreports.engine.JasperReport;
import net.sf.jasperreports.engine.util.JRLoader;
import net.sf.jasperreports.view.JasperViewer;
import org.apache.log4j.Logger;

/**
```

```
 * Clase que contiene la lógica de negocios y rutinas necesarias para la
 * ejecución de reportes Jasper
 * @author Fas
 */
public class GestorReportesBO {

    private JasperReport reporte;
    private JasperPrint impresion;
    private static Logger logger = Logger.getLogger(GestorReportesBO.class);

    public GestorReportesBO() {
    }

    public JasperPrint obtenerImpresion(String nombreReporte, Map parametros)
throws ErrorAplicativo {

        try {
            //reporte = (JasperReport) JRLoader.loadObject(new
File(getClass().getResource("/com/jabluecatch/recursos/reportes/" + nombreReporte +
".jasper").toURI()));
            File archivoReporte = this.obtenerReporte(nombreReporte);
            reporte = (JasperReport) JRLoader.loadObject(archivoReporte);
            archivoReporte.delete();
            impresion = GestorReportesDAO.obtenerImpresion(reporte, parametros);

        } catch (JRException jre) {
            logger.warn("obtenerImpresion - Error al obtener el Reporte - " +
jre.getMessage());
            throw new ErrorAplicativo("GestorReportesBO", "obtenerImpresion",
"Error al obtener el reporte", jre.getMessage());
        }
        return impresion;

    }

    public File obtenerReporte(String nombreReporte) throws ErrorAplicativo {

        InputStream is =
getClass().getResourceAsStream("/com/jabluecatch/recursos/reportes/" +
nombreReporte + ".jasper");
        File result = new File(System.getProperty("java.io.tmpdir") +
"tmp.jasper");

        try {
            OutputStream os = new FileOutputStream(result);
            byte buffer[] = new byte[1024];
            int longitud;

            while ((longitud = is.read(buffer)) > 0) {
                os.write(buffer, 0, longitud);
            }

            os.close();
            is.close();

        } catch (FileNotFoundException jre) {
            logger.warn("obtenerReporte - Error al obtener el Reporte - " +
jre.getMessage());
```

```
            throw new ErrorAplicativo("GestorReportesBO", "obtenerImpresion",
"Error al obtener el reporte", jre.getMessage());
        } catch (IOException jre) {
            logger.warn("obtenerReporte - Error al obtener el Reporte - " +
jre.getMessage());
            throw new ErrorAplicativo("GestorReportesBO", "obtenerImpresion",
"Error al obtener el reporte", jre.getMessage());
        }

        return result;
    }

    public void mostrarReporte(String nombreReporte, boolean exportarPDF, Map
parametros) throws ErrorAplicativo {
        JasperViewer.viewReport(this.obtenerImpresion(nombreReporte, parametros),
false);
        if (exportarPDF) {
            try {
                ParametroGeneral rutaReportes =
ParametroGeneralBO.buscarPorParametro("RUTA_REPORTES");
                File directorioReportes = new File(rutaReportes.getValor());
                if (!directorioReportes.exists()) {
                    directorioReportes.mkdir();
                }
                directorioReportes = null;
                String instancia =
FormateadorFecha.aFileDateTimeString(Calendar.getInstance().getTime());
                logger.debug("mostrarReporte - Ruta a exportar: " +
rutaReportes.getValor() + "/" + nombreReporte + ".pdf");
                JasperExportManager.exportReportToPdfFile(this.impresion,
rutaReportes.getValor() + "\\" + nombreReporte + instancia + ".pdf");
            } catch (JRException jre) {
                logger.warn("mostrarReporte - Error al exportar el Reporte - " +
jre.getMessage());
                throw new ErrorAplicativo("GestorReportesBO", "mostrarReporte",
"Error al exportar el reporte", jre.getMessage());
            }
        }
    }
}
```

4.1.4.8 ParametroGeneralBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.ParametroGeneralDAO;
import com.jabluecatch.entidades.ParametroGeneral;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.SQLException;
import org.apache.log4j.Logger;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
```

```
 * para la entidad PARAMETROS_GENERALES
 * @author Fas
 */
public class ParametroGeneralBO {

    private static Logger logger = Logger.getLogger(ParametroGeneralBO.class);

    public static void insertar(ParametroGeneral p) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();
        ParametroGeneralDAO.insertar(p);
        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("ParametroGeneralBO", "insertar", "Error al
ejecutar la transacción", sqle.getMessage());
        }

    }

    public static void actualizar(ParametroGeneral p) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();
        ParametroGeneralDAO.actualizar(p);
        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("ParametroGeneralBO", "insertar", "Error al
ejecutar la transacción", sqle.getMessage());
        }
    }

    public static ParametroGeneral buscarPorParametroYEstado(String parametro,
String estado) throws ErrorAplicativo {
        return ParametroGeneralDAO.buscarPorParametroYEstado(parametro, estado);
    }
}
```

4.1.4.9 RolBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.RolDAO;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
 * para la entidad ROLES
 * @author Fas
 */
```

```
public class RolBO {

    public static Vector buscarPorEstado(String estado) throws ErrorAplicativo {
        return RolDAO.buscarPorEstado(estado);
    }
}
```

4.1.4.10 UsuarioBO

```
package com.jabluecatch.bo;

import com.jabluecatch.dao.UsuarioDAO;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.Conexion;
import com.jabluecatch.util.ErrorAplicativo;
import java.sql.Connection;
import java.sql.SQLException;
import org.apache.log4j.Logger;

/**
 * Clase que contiene la lógica de negocios y validaciones contra datos
 * para la entidad USUARIOS
 * @author Fausto Almeida
 */
public class UsuarioBO {

    private static Logger logger = Logger.getLogger(UsuarioBO.class);

    public static Usuario validarInicioSesion(String nombreUsuario, String clave)
throws ErrorAplicativo {

        Usuario result = null;
        //Verificar que el usuario exista
        result = UsuarioDAO.buscarPorUsuario(nombreUsuario);

        if (result == null) {
            //Usuario no existe
            logger.info("validarInicioSesion - Usuario no existe");
            throw new ErrorAplicativo("UsuarioBO", "validarInicioSesion", "Usuario
no existe", null);
        }

        Usuario usuarioCompara = UsuarioDAO.buscarPorUsuarioYClave(nombreUsuario,
clave);
        if (usuarioCompara == null) {
            //Clave incorrecta
            logger.info("validarInicioSesion - Clave incorrecta ");
            throw new ErrorAplicativo("UsuarioBO", "validarInicioSesion", "Clave
incorrecta", null);
        }
        //Login correcto
        logger.info("validarInicioSesion - Login correcto <" + nombreUsuario +
">");

        return result;
```

```
    }

    public static void insertar(Usuario uNuevo) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        //Verificar que no exista
        Usuario u = UsuarioDAO.buscarPorUsuario(uNuevo.getUsuario());
        if (u != null) {
            logger.info("insertar - Usuario ya existe en la base de datos ");
            throw new ErrorAplicativo("UsuarioBO", "insertar", "Usuario ya existe
en la base de datos", null);
        }

        u = UsuarioDAO.buscarPorIdentificacion(uNuevo.getIdentificacion());
        if (u != null) {
            logger.info("insertar - Identificacion ya existe en la base de datos
");
            throw new ErrorAplicativo("UsuarioBO", "insertar", "Identificacion ya
existe en la base de datos", null);
        }

        UsuarioDAO.insertar(uNuevo);
        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("insertar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("UsuarioBO", "insertar", "Error al ejecutar
la transacción", sqle.getMessage());
        }
    }

    public static void actualizar(Usuario uNuevo) throws ErrorAplicativo {

        Connection c = Conexion.getConexion();

        //Verificar que exista
        Usuario u = UsuarioDAO.buscarPorUsuario(uNuevo.getUsuario());
        if (u == null) {
            logger.info("actualizar - Usuario no existe en la base de datos ");
            throw new ErrorAplicativo("UsuarioBO", "actualizar", "Usuario no existe
en la base de datos", null);
        }

        UsuarioDAO.actualizar(uNuevo);
        try {
            c.commit();
        } catch (SQLException sqle) {
            logger.warn("actualizar - Error al ejecutar la transacción - " +
sqle.getMessage());
            throw new ErrorAplicativo("UsuarioBO", "actualizar", "Error al ejecutar
la transacción", sqle.getMessage());
        }
    }

    public static Usuario buscarPorUsuario(String usuario) throws ErrorAplicativo {
        return UsuarioDAO.buscarPorUsuario(usuario);
```

```
    }
}
```

4.1.5 Paquete com.jabluecatch.comm

4.1.5.1 ClienteOBEX

```
package com.jabluecatch.comm;

import com.jabluecatch.bo.BitacoraCampaniaBO;
import com.jabluecatch.bo.DispositivoBO;
import com.jabluecatch.dto.Tarea;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.BitacoraCampania;
import com.jabluecatch.util.ErrorAplicativo;
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.util.Calendar;
import javax.bluetooth.BluetoothConnectionException;
import javax.microedition.io.Connector;
import javax.obex.ClientSession;
import javax.obex.HeaderSet;
import javax.obex.Operation;
import javax.obex.ResponseCodes;
import org.apache.log4j.Logger;

/**
 * Clase que se encarga de comenzar y finalizar la tarea de transferencia
 * de contenido hacia el dispositivo correspondiente
 * @author Fausto Almeida
 */
public class ClienteOBEX extends Thread {

    private PantallaPrincipal padre;
    private Tarea tarea;
    private BitacoraCampania bitacora;
    private boolean finalizado;
    private static Logger logger = Logger.getLogger(ClienteOBEX.class);

    public ClienteOBEX(PantallaPrincipal padre, Tarea tarea) {
        this.padre = padre;
        this.tarea = tarea;
        this.finalizado = false;
    }

    @Override
    public void run() {
```

```
        /*
         * Proceso hilado que se encarga de la ejecución de la tarea (envío
         * del contenido multimedia) hacia el dsipositivo Bluetooth
         * correspondiente.
         */

        try {

            logger.info("run: Verificando que el archivo a enviar exista.");

            File f = new File(this.tarea.getArchivoAEnviar().getRuta());
            /*
             * Verificar que el archivo a enviar exista.
             */
            if (!f.exists()) {
                logger.warn("run - El archivo <" +
this.tarea.getArchivoAEnviar().getRuta().replace("\\\\", "\\") + "> no existe.");
                //Guardar la bitácora en la base
                bitacora = new
BitacoraCampania(this.tarea.getArchivoAEnviar().getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(),
this.tarea.getReceptor().getIdDispositivo(), Calendar.getInstance().getTime(), "X",
"Transmisión fallida porque el archivo a enviar no existe.", "A");
                try {
                    BitacoraCampaniaBO.insertar(bitacora);
                } catch (ErrorAplicativo ea) {
                    logger.warn(ea.getErrorTecnico());
                }
                this.tarea.setEstado("X");
                this.padre.actualizarRegistroMonitor(this.tarea);
                //this.tarea.getReceptor()
                this.finalizado = true;
                return;
            }

            logger.info("run: Estableciendo conexión con el dispositivo: " +
this.tarea.getReceptor().getDireccionBt() + " con nombre <" +
this.tarea.getReceptor().getNombreBt() + ">");

            ClientSession sesion = (ClientSession)
Connector.open(this.tarea.getUrlServidor());
            HeaderSet hsRespuestaConexion = sesion.connect(null);
            if (hsRespuestaConexion.getResponseCode() !=
ResponseCodes.OBEX_HTTP_OK) {
                logger.warn("run: Conexión negada  el dispositivo: " +
this.tarea.getReceptor().getDireccionBt() + " con nombre <" +
this.tarea.getReceptor().getNombreBt() + ">");
                return;
            }

            /*
             * Guardar la bitácora en la base.
             */
            bitacora = new
BitacoraCampania(this.tarea.getArchivoAEnviar().getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(),
```

```
this.tarea.getReceptor().getIdDispositivo(), Calendar.getInstance().getTime(), "I",
"Inicio transmisión", "A");
            try {
                BitacoraCampaniaBO.insertar(bitacora);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
            }
            this.tarea.setEstado("I");
            this.padre.actualizarRegistroMonitor(this.tarea);


            HeaderSet hsOperacion = sesion.createHeaderSet();
            logger.debug(this.tarea.getReceptor().getDireccionBt() + " - " +
this.tarea.getArchivoAEnviar().getRuta());

            String nombreArchivo =
this.tarea.getArchivoAEnviar().getRuta().substring(this.tarea.getArchivoAEnviar().g
etRuta().lastIndexOf("\\") + 1, this.tarea.getArchivoAEnviar().getRuta().length());
            logger.debug(this.tarea.getReceptor().getDireccionBt() + " - " +
nombreArchivo);

            hsOperacion.setHeader(HeaderSet.NAME, nombreArchivo);

            logger.debug("Creando operacion de PUT");
            Operation operacionPUT = sesion.put(hsOperacion);

            logger.debug("Inicio de carga de archivo para calcular el numero de
paquetes a enviar. Archivo: " + this.tarea.getArchivoAEnviar().getRuta() + "
Destinatario: " + this.tarea.getReceptor().getDireccionBt());
            BufferedInputStream bis = new BufferedInputStream(new
FileInputStream(this.tarea.getArchivoAEnviar().getRuta()));

            /*
             * Bucle para leer de un fichero y contar las tramas.
             */
            byte[] bufer = new byte[1000];
            int cantidadBytesLeidos = bis.read(bufer);
            double cantidadPaquetes = 0;
            while (cantidadBytesLeidos > 0) {
                cantidadPaquetes++;
                cantidadBytesLeidos = bis.read(bufer);
            }
            logger.debug("Finalización de carga de archivo para calcular el numero
de paquetes a enviar. Archivo: " + this.tarea.getArchivoAEnviar().getRuta() + "
Destinatario: " + this.tarea.getReceptor().getDireccionBt());
            bis.close();

            logger.debug("Inicio de envío de archivo. Archivo: " +
this.tarea.getArchivoAEnviar().getRuta() + " Destinatario: " +
this.tarea.getReceptor().getDireccionBt());
            bis = new BufferedInputStream(new
FileInputStream(this.tarea.getArchivoAEnviar().getRuta()));
            OutputStream os = operacionPUT.openOutputStream();

            bufer = new byte[1000];
            cantidadBytesLeidos = bis.read(bufer);
            double paqueteActual = 1;
            int porcentajeCompletado = 0;
```

```
            logger.debug("Num paquetes totales: " + cantidadPaquetes + " Archivo: "
+ this.tarea.getArchivoAEnviar().getRuta() + " Destinatario: " +
this.tarea.getReceptor().getDireccionBt());
            /*
             * Bucle para enviar cada trama del archivo hacia el dispositivo..
             */
            while (cantidadBytesLeidos > 0) {
                porcentajeCompletado = (int) (paqueteActual / cantidadPaquetes *
100);
                //logger.debug("Porcentaje: " + porcentajeCompletado + " para: "
+this.tarea.getReceptor().getDireccionBt());
                if ((porcentajeCompletado % 16 == 0) || porcentajeCompletado ==
100) {

this.tarea.setPorcentajeCompletado(String.valueOf(porcentajeCompletado));
                    this.padre.actualizarRegistroMonitor(this.tarea);
                }
                os.write(bufer);
                cantidadBytesLeidos = bis.read(bufer);
                paqueteActual++;


            }

            logger.debug("Finalización de envío de archivo. Archivo: " +
this.tarea.getArchivoAEnviar().getRuta() + " Destinatario: " +
this.tarea.getReceptor().getDireccionBt());
            bis.close();

            os.close();
            operacionPUT.close();
            sesion.disconnect(null);
            sesion.close();

            /*
             * Guardar la bitácora en la base.
             */
            bitacora = new
BitacoraCampania(this.tarea.getArchivoAEnviar().getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(),
this.tarea.getReceptor().getIdDispositivo(), Calendar.getInstance().getTime(), "E",
"Transmisión exitosa", "A");
            try {
                BitacoraCampaniaBO.insertar(bitacora);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
            }
            this.tarea.setEstado("E");
            this.padre.actualizarRegistroMonitor(this.tarea);

            /*
             * Actualizar bloqueo de equipo.
             */
            this.tarea.getReceptor().setEstadoBloqueado("S");

this.tarea.getReceptor().setFechaUltimoBloqueo(Calendar.getInstance().getTime());
            try {
```

```
                DispositivoBO.actualizar(this.tarea.getReceptor());
            } catch (ErrorAplicativo ea) {
                logger.warn("run - Error al actualizar el estado de bloqueo del
dispositivo: " + this.tarea.getReceptor().getDireccionBt() + " - " +
ea.getErrorTecnico());
            }
            logger.info("deviceDiscovered - Dispositivo: " +
this.tarea.getReceptor().getDireccionBt() + " ha entrado en la cola de bloqueo");
            this.finalizado = true;

        } catch (BluetoothConnectionException bce) {
            logger.warn("run - El cliente no acepto archivo: " +
this.tarea.getArchivoAEnviar().getRuta() + " hacia el dispositivo: " +
this.tarea.getReceptor().getDireccionBt());
            logger.warn(bce.toString());
            //Guardar la bitácora en la base
            bitacora = new
BitacoraCampania(this.tarea.getArchivoAEnviar().getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(),
this.tarea.getReceptor().getIdDispositivo(), Calendar.getInstance().getTime(), "C",
"Transmisión cancelada por el usuario", "A");
            try {
                BitacoraCampaniaBO.insertar(bitacora);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
            }
            this.tarea.setEstado("C");
            this.padre.actualizarRegistroMonitor(this.tarea);

            /*
             * Actualizar bloqueo de equipo.
             */
            this.tarea.getReceptor().setEstadoBloqueado("S");

this.tarea.getReceptor().setFechaUltimoBloqueo(Calendar.getInstance().getTime());
            try {
                DispositivoBO.actualizar(this.tarea.getReceptor());
            } catch (ErrorAplicativo ea) {
                logger.warn("run - Error al actualizar el estado de bloqueo del
dispositivo: " + this.tarea.getReceptor().getDireccionBt() + " - " +
ea.getErrorTecnico());
            }
            logger.info("deviceDiscovered - Dispositivo: " +
this.tarea.getReceptor().getDireccionBt() + " ha entrado en la cola de bloqueo");
            this.finalizado = true;
            return;
        } catch (IOException ioe) {
            logger.warn("run - Error en el hilo de envío del archivo: " +
this.tarea.getArchivoAEnviar().getRuta() + " hacia el dispositivo: " +
this.tarea.getReceptor().getDireccionBt());
            logger.warn(ioe.toString());
            /*
             * Guardar la bitácora en la base.
             */
            bitacora = new
BitacoraCampania(this.tarea.getArchivoAEnviar().getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(),
```

```
this.tarea.getReceptor().getIdDispositivo(), Calendar.getInstance().getTime(), "X",
"Transmisión fallida debido a un error inesperado", "A");
            try {
                BitacoraCampaniaBO.insertar(bitacora);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
            }
            this.tarea.setEstado("X");
            this.padre.actualizarRegistroMonitor(this.tarea);
            this.finalizado = true;
            return;
        }
    }

    /**
     * @return the finalizado
     */
    public boolean isFinalizado() {
        return finalizado;
    }
}
```

4.1.5.2 Despachador

```
package com.jabluecatch.comm;

import com.jabluecatch.bo.BitacoraCampaniaBO;
import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.dto.Tarea;
import com.jabluecatch.entidades.BitacoraCampania;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Calendar;
import java.util.List;
import java.util.Vector;
import org.apache.log4j.Logger;

/**
 * Clase que se encarga de manejar todos los hilos de ejecución de tareas
 * @author Fausto Almeida
 */
public class Despachador extends Thread {

    private PantallaPrincipal padre;
    private DetectorDispositivosRemotos detectorDispositivosRemotos;
    private boolean continuar;
    private boolean debeInformarPausa;
    private Campania campania;
    private Vector<DetalleCampania> detallesCampania;
    private static Logger logger = Logger.getLogger(Despachador.class);

    public Despachador(PantallaPrincipal padre, Campania campania, Vector
detallesCampania) {
```

```
        this.padre = padre;
        this.cambiarCampania(campania, detallesCampania);
        this.continuar = false;
        this.debeInformarPausa = false;
        this.detectorDispositivosRemotos = new DetectorDispositivosRemotos(padre);

    }

    @Override
    @SuppressWarnings("static-access")
    public void run() {

        while (true) {
            while (continuar) {

                /*
                 * Si la campaña es operativa, asegurarse de que siga activa.
                 */
                if ( !campania.getTipo().equals("O") ? false :
campania.getFechaFin().before(Calendar.getInstance().getTime()) ) {
                    try {
                        CampaniaBO.conciliarFase(campania.getIdCampania());
                    } catch (ErrorAplicativo ea) {
                        logger.warn(ea.getErrorTecnico());

                    }
                    this.setContinuar(false);
                    this.padre.reiniciarBotones();
                    this.padre.mostrarMensaje("La campaña actual se ha detenido");
                    continue;
                }

                this.debeInformarPausa = true;
                /*
                 * Iniciar la detección de dispositivos.
                 */
                try {

this.detectorDispositivosRemotos.detectar(this.detallesCampania);
                } catch (ErrorAplicativo ea) {
                    this.setContinuar(false);
                    logger.warn(ea.getErrorTecnico());
                    this.padre.reiniciarBotones();
                    this.padre.mostrarMensaje("La campaña actual se ha detenido");
                }

                /*
                 * Recorrer el vector de tareas, y ejecutar cada una con un hilo
                 */
                logger.debug("Inicia a recorrer las tareas.");
                List<Tarea> tareas = this.padre.getMonitor();
                Vector<ClienteOBEX> hilosTareas = new Vector();
                DetectorServicio ds;
                String url;
                /*
                 * De cada tarea generada (dispositivo encontrado), detectar si
                 * posee el servicio OBEX necesario.
```

```
                 */
                for (Tarea t : tareas) {

                    if (t.getEstado().equals("P")) {
                        url = null;

                        ds = new DetectorServicio(t.getServidor());
                        try {
                            url = ds.buscarServicio();
                        } catch (ErrorAplicativo ea) {
                            logger.warn(ea.getErrorTecnico());
                        }

                        if (url == null || url.equals("")) {
                            logger.info("No se ha encontrado el servicio en el
dispositivo " + t.getReceptor().getDireccionBt() + ". Envío fallido.");
                            /*
                             * Guardar la bitácora en la base.
                             */
                            BitacoraCampania bc = new
BitacoraCampania(t.getArchivoAEnviar().getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(), t.getReceptor().getIdDispositivo(),
Calendar.getInstance().getTime(), "X", "No se ha encontrado el servicio en el
dispositivo " + t.getReceptor().getDireccionBt() + ". Envío fallido.", "A");
                            try {
                                bc = BitacoraCampaniaBO.insertar(bc);
                            } catch (ErrorAplicativo ea) {
                                logger.warn(ea.getErrorTecnico());
                                return;
                            }
                            t.setEstado("X");
                            this.padre.actualizarRegistroMonitor(t);
                        } else {
                            logger.debug("Tarea con: " +
t.getReceptor().getDireccionBt() + " iniciada.");
                            t.setUrlServidor(url);
                            ClienteOBEX hiloTarea = new ClienteOBEX(this.padre, t);
                            hilosTareas.add(hiloTarea);

                        }
                    }
                }
                /*
                 * Recorrer iterativamente mi vector de hilos y comenzar a
                 * lanzar uno por uno.
                 */
                for (ClienteOBEX c : hilosTareas) {
                    c.start();
                    try {
                        Thread.currentThread().sleep(2000);
                    } catch (InterruptedException ie) {
                        logger.debug("run - Error al intentar pausar la ejecución
para el lanzamiento de hilos");
                    }
                }
                /*
                 * Recorrer iterativamente mi vector de hilos y salir del
                 * bucle solo cuando el estado de todos este en terminado.
```

```
                 */
                boolean terminados = false;
                while (!terminados) {
                    terminados = true;
                    for (ClienteOBEX c : hilosTareas) {
                        /*
                         * Si encuentro alguino diferente de terminado.
                         */
                        if (!c.isFinalizado()) {
                            terminados = false;
                        }
                    }
                }
                logger.debug("Todas las tareas han finalizado.");
            }
            if (debeInformarPausa) {
                logger.info("btnIniciarActionPerformed - Detenido proceso de envío
de publicidad.");
                this.debeInformarPausa = false;
            }
        }
    }

    public void cambiarCampania(Campania campania, Vector detallesCampania) {
        this.campania = campania;
        this.detallesCampania = detallesCampania;
    }

    /**
     * @return the continuar
     */
    public boolean isContinuar() {
        return continuar;
    }

    /**
     * @param continuar the continuar to set
     */
    public void setContinuar(boolean continuar) {
        this.continuar = continuar;
    }
}
```

4.1.5.3 DetectorDispositivosRemotos

```
package com.jabluecatch.comm;

import com.jabluecatch.bo.BitacoraCampaniaBO;
import com.jabluecatch.bo.DispositivoBO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.BitacoraCampania;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.entidades.Dispositivo;
import com.jabluecatch.dto.Tarea;
import com.jabluecatch.util.ErrorAplicativo;
```

```
import com.jabluecatch.util.ValidadorNombreDispositivo;
import java.io.IOException;
import java.util.Calendar;
import java.util.Vector;
import javax.bluetooth.BluetoothStateException;
import javax.bluetooth.DeviceClass;
import javax.bluetooth.DiscoveryAgent;
import javax.bluetooth.DiscoveryListener;
import javax.bluetooth.LocalDevice;
import javax.bluetooth.RemoteDevice;
import javax.bluetooth.ServiceRecord;
import org.apache.log4j.Logger;

/**
 * Clase que se encarga de detectar dispositivos Bluetooth y atar cada
 * dispositivo con una url y un archivo a enviar, generando así una tarea.
 * @author Fausto Almeida
 */
public class DetectorDispositivosRemotos implements DiscoveryListener {

    private PantallaPrincipal padre;
    private Vector<Dispositivo> dispositivosDetectados;
    private final Object eventoDeteccionCompletada = new Object();
    private Vector<DetalleCampania> detallesCampania;
    private static Logger logger =
Logger.getLogger(DetectorDispositivosRemotos.class);

    public DetectorDispositivosRemotos() {
    }

    public DetectorDispositivosRemotos(PantallaPrincipal padre) {
        this.padre = padre;
    }

    public void iniciar() throws ErrorAplicativo {
        try {
            LocalDevice.getLocalDevice().getDiscoveryAgent();
        } catch (BluetoothStateException bse) {
            logger.warn("run - Error: " + bse.getMessage());
            throw new ErrorAplicativo("DetectorDispositivosRemotos", "iniciar", "No
se encuentra el dispositivo Bluetooth, favor conectarlo", bse.getMessage());
        }
    }

    public void detectar(Vector detallesCampania) throws ErrorAplicativo {

        this.detallesCampania = detallesCampania;
        this.dispositivosDetectados = new Vector();

        /*
         * Inicia el proceso de detección que por convención y metodología
         * debe tratarse como un proceso sincronizado.
         */
        synchronized (this.eventoDeteccionCompletada) {
            try {
                boolean iniciado =
LocalDevice.getLocalDevice().getDiscoveryAgent().startInquiry(DiscoveryAgent.GIAC,
this);
```

```
                if (iniciado) {
                    logger.info("detectar - Detección de dispositivos
INICIADA...");
                    this.eventoDeteccionCompletada.wait();
                    logger.info(getDispositivosDetectados().size() + "
dispositivo(s) encontrado(s)");
                }
            } catch (BluetoothStateException bse) {
                logger.warn("run - Error: " + bse.getMessage());
                throw new ErrorAplicativo("DetectorDispositivosRemotos",
"detectar", "Error de comunicación", bse.getMessage());
            } catch (InterruptedException ie) {
                logger.warn("run - Error: " + ie.getMessage());
                throw new ErrorAplicativo("DetectorDispositivosRemotos",
"detectar", "Error de interrupción", ie.getMessage());
            }

        }
    }

    public void deviceDiscovered(RemoteDevice rd, DeviceClass dc) {

        /*
         * Método que se lanza cada vez que el proceso detecta un nuevo
         * dispositivo.
         */
        boolean autorizadoParaEnvio = false;
        boolean realizarCambioDeNombre = false;
        /*
         * Llenar los datos del dispositivo.
         */
        Dispositivo d = new Dispositivo();
        d.setDireccionBt(rd.getBluetoothAddress());
        logger.info("deviceDiscovered - Dispositivo encontrado: " +
rd.getBluetoothAddress());
        try {
            d.setNombreBt(rd.getFriendlyName(false).replace("\\", "").replace("/",
""));
            logger.info("deviceDiscovered - Nombre dispositivo encontrado: " +
d.getNombreBt());
        } catch (IOException ioe) {
            d.setNombreBt("N/A");
            logger.warn("deviceDiscovered - Error: " + ioe.getMessage());
        }
        /*
         * Validacion de nombre con caracteres especiales.
         */
        if (!ValidadorNombreDispositivo.esValido(d.getNombreBt())) {
            d.setNombreBt("Dispositivo");
            realizarCambioDeNombre = true;
        }
        d.setEstadoBloqueado("N");
        d.setEstado("A");

        /*
         * Insertar o actualizar el dispositivo.
         */
        try {
```

```
            autorizadoParaEnvio = DispositivoBO.autorizarParaEnvio(d);
            d = DispositivoBO.buscarPorDireccionBt(d.getDireccionBt());
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
        }

        /*
         * De contener el nombre caracteres especiales, se actualizara su nombre.
         */
        if (realizarCambioDeNombre) {
            d.setNombreBt(d.getNombreBt() + " " + d.getIdDispositivo());
            try {
                DispositivoBO.actualizar(d);
                d = DispositivoBO.buscarPorDireccionBt(d.getDireccionBt());
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
            }
        }

        //if (rd.getBluetoothAddress().equals("001DFD9F56F4")) {
        /*
         * Guardar la referencia del dispositivo en nuestro vector.
         */
        if (autorizadoParaEnvio) {
            logger.info("deviceDiscovered - Dispositivo: " +
rd.getBluetoothAddress() + " ha sido autorizado para el envío");
            this.getDispositivosDetectados().addElement(d);
            DetalleCampania archivoAleatorio = this.obtenerArchivoAleatorio();
            /*
             * Cargar el id_dispositivo desde la base.
             */
            try {
                d = DispositivoBO.buscarPorDireccionBt(d.getDireccionBt());
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                return;
            }
            /*
             * Guardar la bitácora en la base.
             */
            BitacoraCampania bc = new
BitacoraCampania(archivoAleatorio.getIdDetalleCampania(),
this.padre.getUsuarioSesion().getIdUsuario(), d.getIdDispositivo(),
Calendar.getInstance().getTime(), "P", "Dispositivo detectado", "A");
            try {
                bc = BitacoraCampaniaBO.insertar(bc);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                return;
            }
            /*
             * Mostrar la tarea en el monitor.
             */
            this.padre.agregarRegistroMonitor(new Tarea(bc.getIdBitacoraCampania(),
rd, d, archivoAleatorio, "P", "N/A"));
            logger.debug("deviceDiscovered - Tarea creada con el código: " +
bc.getIdBitacoraCampania());
        } else {
```

```
            logger.info("deviceDiscovered - Dispositivo: " +
rd.getBluetoothAddress() + " no ha sido autorizado para el envío");
        }
        //}
    }

    public void inquiryCompleted(int i) {
        logger.info("Detección de dispositivos FINALIZADA");
        synchronized (this.eventoDeteccionCompletada) {
            this.eventoDeteccionCompletada.notifyAll();
        }
    }

    private DetalleCampania obtenerArchivoAleatorio() {
        int indiceAleatorio = (int) (Math.random() *
(this.detallesCampania.size()));
        return this.detallesCampania.get(indiceAleatorio);
    }

    public void servicesDiscovered(int i, ServiceRecord[] srs) {
    }

    public void serviceSearchCompleted(int i, int i1) {
    }

    /**
     * @return the dispositivosDetectados
     */
    public Vector<Dispositivo> getDispositivosDetectados() {
        return dispositivosDetectados;
    }
}
```

4.1.5.4 DetectorServicio

```
package com.jabluecatch.comm;

import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;
import javax.bluetooth.BluetoothStateException;
import javax.bluetooth.DataElement;
import javax.bluetooth.DeviceClass;
import javax.bluetooth.DiscoveryListener;
import javax.bluetooth.LocalDevice;
import javax.bluetooth.RemoteDevice;
import javax.bluetooth.ServiceRecord;
import javax.bluetooth.UUID;
import org.apache.log4j.Logger;

/**
 * Clase que se encarga de detectar si el servicio OBEX se encuentra activo
 * en el dispositivo encontrado (servidor)
 * @author Fausto Almeida
 */
public class DetectorServicio implements DiscoveryListener {
```

```
    private static final UUID[] SERVICIOS_BUSCAR = new UUID[]{new UUID(0x1105)};
    private static final int[] ATRIBUTOS_BUSCAR = new int[]{0x0100};
    private RemoteDevice servidor;
    private Vector<String> serviciosEncontrados = new Vector();
    private final Object eventoDeteccionCompletada = new Object();
    private static Logger logger = Logger.getLogger(DetectorServicio.class);

    public DetectorServicio() {
    }

    public DetectorServicio(RemoteDevice servidor) {
        this.servidor = servidor;
    }

    public String buscarServicio() throws ErrorAplicativo {

        this.serviciosEncontrados.clear();

        /*
         * Inicia el proceso de detección que por convención y metodología
         * debe tratarse como un proceso sincronizado.
         */
        synchronized (this.eventoDeteccionCompletada) {
            try {
                logger.debug("Detección de servicios en " +
this.servidor.getBluetoothAddress() + " INICIADA... ");

LocalDevice.getLocalDevice().getDiscoveryAgent().searchServices(ATRIBUTOS_BUSCAR,
SERVICIOS_BUSCAR, this.servidor, this);
                this.eventoDeteccionCompletada.wait();
            } catch (BluetoothStateException bse) {
                logger.warn("DetectorServicio.run - Error: " + bse.getMessage());
                throw new ErrorAplicativo("DetectorServicio", "buscarServicio",
"Error de comunicación", bse.getMessage());
            } catch (InterruptedException ie) {
                logger.warn("DetectorServicio.run - Error: " + ie.getMessage());
                throw new ErrorAplicativo("DetectorServicio", "buscarServicio",
"Error de interrupción", ie.getMessage());
            }
        }
        if (serviciosEncontrados.size() > 0) {
            return serviciosEncontrados.get(0);
        } else {
            return null;
        }
    }

    public void servicesDiscovered(int transID, ServiceRecord[] sr) {

        /*
         * Método que se lanza cada vez que el proceso detecta un conjunto
         * de servicios en el dispositivo.
         */
        for (int i = 0; i < sr.length; i++) {
            String url =
sr[i].getConnectionURL(ServiceRecord.NOAUTHENTICATE_NOENCRYPT, false);
            if (url == null) {
```

```
                continue;
            }
            this.serviciosEncontrados.add(url);
            DataElement nombreServicio = sr[i].getAttributeValue(0x0100);
            logger.debug("URL: " + url);
            if (nombreServicio != null) {
                logger.debug("Servicio " + nombreServicio.getValue() + "
encontrado. URL: " + url);
            } else {
                logger.debug("Servicio encontrado. URL: " + url);
            }
        }
    }

    public void serviceSearchCompleted(int i, int i1) {
        logger.info("Detección de servicios FINALIZADA");
        synchronized (eventoDeteccionCompletada) {
            eventoDeteccionCompletada.notifyAll();
        }
    }

    public void deviceDiscovered(RemoteDevice rd, DeviceClass dc) {
    }

    public void inquiryCompleted(int i) {
    }
}
```

4.1.6 Paquete com.jabluecatch.util

4.1.6.1 Conexión

```
package com.jabluecatch.util;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import org.apache.log4j.Logger;

/**
 * Clase necesaria para la conexión de base de datos de acuerdo al patrón de
 * diseño: Singleton
 * @author Fas
 */
public class Conexion {

    private static Connection conexion = null;
    private static Connection conexionGenerica = null;
    private static String bd = "bluecatch";
    private static String usuario = "root";
```

```
    private static String clave = "mysql";
    private static String url = "jdbc:mysql://localhost/" + bd;
    private static String urlGenerica = "jdbc:mysql://localhost/";
    private static Logger logger = Logger.getLogger(Conexion.class);

    public static Connection getConexion() throws ErrorAplicativo {
        if (conexion == null) {
            conexionGenerica = null;
            inicializaConexion();
        }
        return conexion;
    }

    protected static void inicializaConexion() throws ErrorAplicativo {
        try {
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            conexion = DriverManager.getConnection(url, usuario, clave);
            conexion.setAutoCommit(false);
        } catch (SQLException sqle) {
            logger.warn("inicializaConexion - Hubo un problema al intentar
conectarse con la base de datos - " + sqle.getMessage());
            throw new ErrorAplicativo("Conexion", "inicializaConexion", "Hubo un
problema al intentar conectarse con la base de datos", sqle.getMessage());
        } catch (Exception e) {
            logger.warn("inicializaConexion - Hubo un problema al intentar
conectarse con la base de datos - " + e.getMessage());
            throw new ErrorAplicativo("Conexion", "inicializaConexion", "Hubo un
problema al intentar conectarse con la base de datos", e.getMessage());
        }
        logger.info("inicializaConexion - Sesión iniciada con la base de datos");
    }

    public static Connection getConexionGenerica() throws ErrorAplicativo {
        if (conexionGenerica == null) {
            conexion = null;
            inicializaConexionGenerica();
        }
        return conexionGenerica;
    }

    protected static void inicializaConexionGenerica() throws ErrorAplicativo {
        try {
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            conexionGenerica = DriverManager.getConnection(urlGenerica, usuario,
clave);
            conexionGenerica.setAutoCommit(false);
        } catch (SQLException sqle) {
            logger.warn("inicializaConexionGenerica - Hubo un problema al intentar
conectarse con la base de datos - " + sqle.getMessage());
            throw new ErrorAplicativo("Conexion", "inicializaConexionGenerica",
"Hubo un problema al intentar conectarse con la base de datos", sqle.getMessage());
        } catch (Exception e) {
            logger.warn("inicializaConexionGenerica - Hubo un problema al intentar
conectarse con la base de datos - " + e.getMessage());
            throw new ErrorAplicativo("Conexion", "inicializaConexionGenerica",
"Hubo un problema al intentar conectarse con la base de datos", e.getMessage());
        }
```

```
        logger.info("inicializaConexionGenerica - Sesión iniciada con la base de
datos");
    }
}
```

4.1.6.2 ErrorAplicativo

```
package com.jabluecatch.util;

/**
 * Clase que representa un error generado en cualquier instancia en la
 * aplicación y que será propagado hacia la capa más externa (vista)
 * @author Fausto Almeida
 */
public class ErrorAplicativo extends Exception {

    private String clase;
    private String metodo;
    private String tecnico;

    public ErrorAplicativo(String clase, String metodo, String message, String
tecnico) {
        super(message);
        this.clase = clase;
        this.metodo = metodo;
        this.tecnico = tecnico;
    }

    public String getError() {
        return this.getMessage();
    }

    public String getErrorTecnico() {
        return this.clase + "." + this.metodo + " - " + this.tecnico;
    }

    /**
     * @return the clase
     */
    public String getClase() {
        return clase;
    }

    /**
     * @param clase the clase to set
     */
    public void setClase(String clase) {
        this.clase = clase;
    }

    /**
     * @return the metodo
     */
    public String getMetodo() {
        return metodo;
```

```
    }

    /**
     * @param metodo the metodo to set
     */
    public void setMetodo(String metodo) {
        this.metodo = metodo;
    }

    /**
     * @return the tecnico
     */
    public String getTecnico() {
        return tecnico;
    }

    /**
     * @param tecnico the tecnico to set
     */
    public void setTecnico(String tecnico) {
        this.tecnico = tecnico;
    }
}
```

4.1.6.3 FormateadorDocumento

```
package com.jabluecatch.util;

import java.util.Vector;
import javax.swing.text.AttributeSet;
import javax.swing.text.BadLocationException;
import javax.swing.text.PlainDocument;

/**
 * Clase que me permite cambiar el formato de ingreso de las cadenas de
 * caracteres en los componentes de entrada de texto de la interfaz de usuario
 * @author Fausto Almeida
 */
public class FormateadorDocumento extends PlainDocument {

    private int longitudMaxima;
    private boolean conLongitudMaxima;
    private int modo;
    private Vector vecLetras;
    private Vector vecNumeros;

    public FormateadorDocumento(int longitudMaxima, int modo) {
        super();
        this.longitudMaxima = longitudMaxima;
        this.conLongitudMaxima = true;
        this.modo = modo;
        vecLetras = new Vector();
        for (int i = 97; i < 123; i++) {
            vecLetras.addElement((char) i);
        }
```

```
        for (int i = 65; i < 91; i++) {
            vecLetras.addElement((char) i);
        }
        vecLetras.addElement('ñ');
        vecLetras.addElement('Ñ');
        vecNumeros = new Vector();
        for (int i = 48; i < 58; i++) {
            vecNumeros.addElement((char) i);
        }
    }

    public void setLongitudMaxima(int longitudMaxima) {
        this.longitudMaxima = longitudMaxima;
        this.conLongitudMaxima = true;
    }

    public void setLongitudMaxima(boolean conLongitudMaxima) {
        this.conLongitudMaxima = conLongitudMaxima;
    }

    @Override
    public void insertString(int inicio, String strCadena, AttributeSet
astAtributos) throws BadLocationException {
        //Convertir a arreglo la cadena ingresada
        char[] cadena;
        cadena = strCadena.toCharArray();

        //Recorre todos los caracteres ingresados
        //for (int i = 0; i < cadena.length; i++) {
        for (int i = cadena.length - 1; i >= 0; i--) {
            if (this.modo == 1) { //Solo numeros
                if (this.conLongitudMaxima) {
                    if (vecNumeros.contains(cadena[i])
                            && this.getContent().length() <= this.longitudMaxima) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                } else {
                    if (vecNumeros.contains(cadena[i])) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                }
            } else if (this.modo == 2) { //Solo letras minusculas y numeros
                if (this.conLongitudMaxima) {
                    if (this.getContent().length() <= this.longitudMaxima) {
                        if ((vecLetras.contains(cadena[i])
                                || vecNumeros.contains(cadena[i]))
                                && cadena[i] != 'ñ'
                                && cadena[i] != 'Ñ') {
                            super.insertString(inicio,
                                    String.valueOf(cadena[i]).toLowerCase(),
                                    astAtributos);
                        }
                    }
                } else {
                    if ((vecLetras.contains(cadena[i])
                            || vecNumeros.contains(cadena[i]))
```

```
                            && cadena[i] != 'ñ'
                            && cadena[i] != 'Ñ') {
                        super.insertString(inicio,
                                String.valueOf(cadena[i]).toLowerCase(),
                                astAtributos);
                    }
                }
            } else if (this.modo == 3) { ////Solo letras, Numeros (1-9) y Espacios
en blanco
                if (this.conLongitudMaxima) {
                    if (this.getContent().length() <= this.longitudMaxima) {
                        if (vecLetras.contains(cadena[i])
                                || vecNumeros.contains(cadena[i])
                                || Character.isSpaceChar(cadena[i])) {
                            super.insertString(inicio,
                                    String.valueOf(cadena[i]),
                                    astAtributos);
                        }
                    }
                } else {
                    if (vecLetras.contains(cadena[i])
                            || vecNumeros.contains(cadena[i])
                            || Character.isSpaceChar(cadena[i])) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                }
            } else if (this.modo == 4) { ////Solo numeros y "/", para fechas
                if (this.conLongitudMaxima) {
                    if ((vecNumeros.contains(cadena[i]) || (cadena[i] == '/'))
                            && this.getContent().length() <= this.longitudMaxima) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                } else {
                    if (vecNumeros.contains(cadena[i]) || (cadena[i] == '/')) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                }
            } else if (this.modo == 5) { ////Solo numeros y ":", para horas
                if (this.conLongitudMaxima) {
                    if ((vecNumeros.contains(cadena[i]) || (cadena[i] == ':'))
                            && this.getContent().length() <= this.longitudMaxima) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                } else {
                    if (vecNumeros.contains(cadena[i]) || (cadena[i] == ':')) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
                    }
                }
            } else {
                if (this.conLongitudMaxima) {
                    if (this.getContent().length() <= this.longitudMaxima) {
                        super.insertString(inicio, String.valueOf(cadena[i]),
                                astAtributos);
```

```
                    }
                } else {

                    super.insertString(inicio, String.valueOf(cadena[i]),
                            astAtributos);
                }
            }
        }
    }
}
```

4.1.6.4 FormateadorFecha

```
package com.jabluecatch.util;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import org.apache.log4j.Logger;

/**
 * Clase que me permite cambiar de formato a las fechas, realizar conversiones
 * de tipo fecha hacia tipo cadena de caracteres, en formatos de base de datos
 * y formatos de lenguaje de programación.
 * @author Fas
 */
public class FormateadorFecha {

    private static final SimpleDateFormat DATE_JAVA = new
SimpleDateFormat("dd/MM/yyyy");
    private static final SimpleDateFormat DATETIME_JAVA = new
SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
    private static final SimpleDateFormat HOUR_JAVA = new SimpleDateFormat("HH");
    private static final SimpleDateFormat MINUTE_JAVA = new SimpleDateFormat("mm");
    private static final SimpleDateFormat DATETIME_FILE = new SimpleDateFormat("-
ddMMyyyy-HHmmss");
    private static Logger logger = Logger.getLogger(FormateadorFecha.class);

    public static String aJavaDateString(Date fecha) {
        return DATE_JAVA.format(fecha);
    }

    public static String aJavaDateTimeString(Date fecha) {
        return DATETIME_JAVA.format(fecha);
    }

    public static String aJavaHourString(Date fecha) {
        return HOUR_JAVA.format(fecha);
    }

    public static String aJavaMinuteString(Date fecha) {
        return MINUTE_JAVA.format(fecha);
    }

    public static String aFileDateTimeString(Date fecha) {
```

```
        return DATETIME_FILE.format(fecha);
    }

    public static Date aJavaDateTimeDate(String fecha) throws ErrorAplicativo {
        try {
            return DATETIME_JAVA.parse(fecha);
        } catch (ParseException pe) {
            logger.warn("FormateadorFecha.aJavaDateDate - Error al convertir la
fecha - " + pe.getMessage());
            throw new ErrorAplicativo("FormateadorFecha", "aJavaDateDate", "La
fecha no está en el formato correcto", pe.getMessage());
        }
    }
}
```

4.1.6.5 Instalador

```
package com.jabluecatch.util;

import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import org.apache.log4j.Logger;

/**
 *
 * @author Fas
 */
public class Instalador {

    private static Logger logger = Logger.getLogger(Instalador.class);

    public static void ejecutar() throws ErrorAplicativo {

        logger.info("ejecutar - Instalación iniciada");

        Connection c = Conexion.getConexionGenerica();

        Statement t;
        try {
            t = c.createStatement();
            t.executeUpdate("SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0");
            t.executeUpdate("SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0");
            t.executeUpdate("SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL'");

            t.executeUpdate("CREATE SCHEMA IF NOT EXISTS `bluecatch` DEFAULT
CHARACTER SET latin1");
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("USE `bluecatch`");

            //Tablas `bluecatch`.`roles`
            t.executeUpdate("DROP TABLE IF EXISTS `bluecatch`.`roles`");
```

```
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS `bluecatch`.`roles` ( "
                    + "`id_rol` INT NOT NULL AUTO_INCREMENT , "
                    + "`rol` VARCHAR(40) NOT NULL , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_rol`) ) "
                    + "ENGINE = InnoDB");
            t.executeUpdate("SHOW WARNINGS");

            //Tabla `bluecatch`.`usuarios`
            t.executeUpdate("DROP TABLE IF EXISTS `bluecatch`.`usuarios` ");
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS `bluecatch`.`usuarios` ( "
                    + "`id_usuario` INT NOT NULL AUTO_INCREMENT , "
                    + "`id_rol` INT NOT NULL , "
                    + "`usuario` VARCHAR(20) NOT NULL , "
                    + "`clave` VARCHAR(32) NOT NULL , "
                    + "`nombres` VARCHAR(60) NOT NULL , "
                    + "`apellidos` VARCHAR(60) NOT NULL , "
                    + "`identificacion` VARCHAR(20) NOT NULL , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_usuario`) , "
                    + "INDEX `usuarios_fk2` (`id_rol` ASC) , "
                    + "CONSTRAINT `usuarios_fk2` "
                    + "FOREIGN KEY (`id_rol` ) "
                    + "REFERENCES `bluecatch`.`roles` (`id_rol` ) "
                    + "ON DELETE NO ACTION "
                    + "ON UPDATE NO ACTION) "
                    + "ENGINE = InnoDB");
            t.executeUpdate("SHOW WARNINGS");

            //Tabla `bluecatch`.`campanias`
            t.executeUpdate("DROP TABLE IF EXISTS `bluecatch`.`campanias` ");
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS `bluecatch`.`campanias` (
"
                    + "`id_campania` INT NOT NULL AUTO_INCREMENT , "
                    + "`campania` VARCHAR(200) NOT NULL , "
                    + "`descripcion` VARCHAR(500) NOT NULL , "
                    + "`tipo` VARCHAR(1) NOT NULL COMMENT
'O=Operacional\nP=Permanente' , "
                    + "`costo_ref_mensaje` FLOAT NOT NULL , "
                    + "`fecha_inicio` DATETIME NOT NULL , "
                    + "`fecha_fin` DATETIME NULL , "
                    + "`fase` VARCHAR(1) NOT NULL COMMENT
'P=Pendiente\nE=Ejecucion\nF=Finalizada' , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_campania`) ) "
                    + "ENGINE = InnoDB");
            t.executeUpdate("SHOW WARNINGS");

            //Tabla `bluecatch`.`detalle_campanias`
            t.executeUpdate("DROP TABLE IF EXISTS `bluecatch`.`detalle_campanias`
");
            t.executeUpdate("SHOW WARNINGS");
```

```
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS
`bluecatch`.`detalle_campanias` ( "
                    + "`id_detalle_campania` INT NOT NULL AUTO_INCREMENT , "
                    + "`id_campania` INT NOT NULL , "
                    + "`ruta` VARCHAR(256) NOT NULL , "
                    + "`tipo` VARCHAR(1) NOT NULL COMMENT 'A=Audio\nI=Imagen' , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_detalle_campania`) , "
                    + "INDEX `detalle_campanias_fk2` (`id_campania` ASC) , "
                    + "CONSTRAINT `detalle_campanias_fk2` "
                    + "FOREIGN KEY (`id_campania` ) "
                    + "REFERENCES `bluecatch`.`campanias` (`id_campania` ) "
                    + "ON DELETE NO ACTION "
                    + "ON UPDATE NO ACTION) "
                    + "ENGINE = InnoDB");
            t.executeUpdate("SHOW WARNINGS");

            //Tabla `bluecatch`.`dispositivos`
            t.executeUpdate("DROP TABLE IF EXISTS `bluecatch`.`dispositivos`");
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS `bluecatch`.`dispositivos`
( "
                    + "`id_dispositivo` INT NOT NULL AUTO_INCREMENT , "
                    + "`direccion_bt` VARCHAR(20) NOT NULL , "
                    + "`nombre_bt` VARCHAR(100) NULL , "
                    + "`estado_bloqueado` VARCHAR(1) NOT NULL COMMENT 'N=No esta
bloqueado\nS=Si esta bloqueado' , "
                    + "`fecha_ultimo_bloqueo` DATETIME NULL , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_dispositivo`) ) "
                    + "ENGINE = InnoDB "
                    + "DEFAULT CHARACTER SET = big5 "
                    + "COLLATE = big5_bin");
            t.executeUpdate("SHOW WARNINGS");

            //Tabla `bluecatch`.`bitacora_campanias`
            t.executeUpdate("DROP TABLE IF EXISTS `bluecatch`.`bitacora_campanias`
");
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS
`bluecatch`.`bitacora_campanias` ( "
                    + "`id_bitacora_campania` INT NOT NULL AUTO_INCREMENT , "
                    + "`id_detalle_campania` INT NOT NULL , "
                    + "`id_usuario` INT NOT NULL , "
                    + "`id_dispositivo_destino` INT NOT NULL , "
                    + "`fecha_registro` DATETIME NOT NULL , "
                    + "`evento` VARCHAR(1) NOT NULL COMMENT 'P=Pendiente\nI=Inicio
transmision\nE=Transmision exitosa\nX=Transmision fallida\nC=Transmision cancelada'
, "
                    + "`observacion` VARCHAR(500) NOT NULL , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_bitacora_campania`) , "
                    + "INDEX `bitacora_campanias_fk3` (`id_usuario` ASC) , "
                    + "INDEX `bitacora_campanias_fk2` (`id_detalle_campania` ASC) ,
"
```

```
                    + "INDEX `bitacora_campanias_fk4` (`id_dispositivo_destino`
ASC) , "
                    + "CONSTRAINT `bitacora_campanias_fk3` "
                    + "FOREIGN KEY (`id_usuario` ) "
                    + "REFERENCES `bluecatch`.`usuarios` (`id_usuario` ) "
                    + "ON DELETE NO ACTION "
                    + "ON UPDATE NO ACTION, "
                    + "CONSTRAINT `bitacora_campanias_fk2` "
                    + "FOREIGN KEY (`id_detalle_campania` ) "
                    + "REFERENCES `bluecatch`.`detalle_campanias`
(`id_detalle_campania` ) "
                    + "ON DELETE NO ACTION "
                    + "ON UPDATE NO ACTION, "
                    + "CONSTRAINT `bitacora_campanias_fk4` "
                    + "FOREIGN KEY (`id_dispositivo_destino` ) "
                    + "REFERENCES `bluecatch`.`dispositivos` (`id_dispositivo` ) "
                    + "ON DELETE NO ACTION "
                    + "ON UPDATE NO ACTION) "
                    + "ENGINE = InnoDB");
            t.executeUpdate("SHOW WARNINGS");

            //Tabla `bluecatch`.`parametros_generales`
            t.executeUpdate("DROP TABLE IF EXISTS
`bluecatch`.`parametros_generales` ");
            t.executeUpdate("SHOW WARNINGS");
            t.executeUpdate("CREATE  TABLE IF NOT EXISTS
`bluecatch`.`parametros_generales` ( "
                    + "`id_parametro_general` INT NOT NULL AUTO_INCREMENT , "
                    + "`parametro` VARCHAR(30) NOT NULL , "
                    + "`descripcion` VARCHAR(500) NOT NULL , "
                    + "`valor` VARCHAR(500) NOT NULL , "
                    + "`tipo` VARCHAR(1) NOT NULL COMMENT 'V=Valor\nM=Mensaje' , "
                    + "`estado` VARCHAR(1) NOT NULL COMMENT 'A=Activo\nI=Inactivo'
, "
                    + "PRIMARY KEY (`id_parametro_general`) ) "
                    + "ENGINE = InnoDB");
            t.executeUpdate("SHOW WARNINGS");

            t.executeUpdate("SET SQL_MODE=@OLD_SQL_MODE");
            t.executeUpdate("SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS");
            t.executeUpdate("SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS");

            // Registros necesarios para la ejecución
            t.executeUpdate("INSERT INTO roles (rol, estado) VALUES
('Administrador', 'A')");
            t.executeUpdate("INSERT INTO roles (rol, estado) VALUES ('Operador',
'A');");
            t.executeUpdate("INSERT INTO usuarios (id_rol, usuario, clave, nombres,
apellidos, identificacion, estado) "
                    + "VALUES (1, 'admin', AES_ENCRYPT('admin','admi'), 'Nombre
Administrador', 'Apellido Administrador', '0000000000', 'A');");

            t.executeUpdate("INSERT INTO parametros_generales (parametro,
descripcion, valor, tipo, estado) "
                    + "VALUES ('TIEMPO_BLOQUEO', "
                    + "'Tiempo que un dispositivo no podrá recibir un archivo de
publicidad luego de haber recibio uno satisfactoriamente', "
                    + "'10', "
```

```
                    + "'V', "
                    + "'A')");
            t.executeUpdate("INSERT INTO parametros_generales (parametro,
descripcion, valor, tipo, estado) "
                    + "VALUES ('RUTA_REPORTES', "
                    + "'Ruta en donde se exportarán los reportes del sistema en
formato PDF', "
                    + "'C:\\\\Reportes BlueCatch', "
                    + "'C', "
                    + "'A');");
            t.executeUpdate("INSERT INTO parametros_generales (parametro,
descripcion, valor, tipo, estado) "
                    + "VALUES ('INSTALADO', "
                    + "'Parametro que determina si la instalacion se realizó
correctamente', "
                    + "'S', "
                    + "'V', "
                    + "'A')");

            c.commit();
            logger.info("ejecutar - Instalación finalizada correctamente");
        } catch (SQLException sqle) {
            logger.warn("ejecutar - Error al ejecutar la instalación - " +
sqle.getMessage());
            throw new ErrorAplicativo("Instalador", "ejecutar", "Error al ejecutar
la instalación", sqle.getMessage());
        }
    }
}
```

4.1.6.6 ModeloTablaCampanias

```
package com.jabluecatch.util;

import com.jabluecatch.dto.ConsultaCampanias;
import java.util.Vector;
import javax.swing.table.AbstractTableModel;

/**
 * Clase que representa el modelo para armar las tablas en donde se muestran
 * instancias de las campañas
 * @author Fausto Almeida
 */
public class ModeloTablaCampanias extends AbstractTableModel {

    Vector<ConsultaCampanias> filas;

    public ModeloTablaCampanias() {
        filas = new Vector();
    }

    public ModeloTablaCampanias(Vector filas) {
        this.filas = filas;
    }
```

```
    public Vector getFilas() {
        return filas;
    }

    public void setFilas(Vector filas) {
        this.filas = filas;
    }

    public int getRowCount() {
        return this.filas.size();
    }

    public int getColumnCount() {
        return 6;
    }

    public Object getValueAt(int rowIndex, int columnIndex) {
        ConsultaCampanias c = this.filas.get(rowIndex);
        Object result = null;
        if (columnIndex == 0) {
            result = c.getIdCampania();
        } else if (columnIndex == 1) {
            result = c.getCampania();
        } else if (columnIndex == 2) {
            result = c.getTipo();
        } else if (columnIndex == 3) {
            result = c.getFechaInicio();
        } else if (columnIndex == 4) {
            result = c.getFechaFin();
        } else if (columnIndex == 5) {
            result = (c.getFase().equals("P") ? "Pendiente" :
(c.getFase().equals("E") ? "Ejecución" : "Finalizada"));
        }
        return result;
    }

    @Override
    public String getColumnName(int columnIndex) {
        String result = null;
        if (columnIndex == 0) {
            result = "Id";
        } else if (columnIndex == 1) {
            result = "Campaña";
        } else if (columnIndex == 2) {
            result = "Tipo";
        } else if (columnIndex == 3) {
            result = "Fecha de inicio";
        } else if (columnIndex == 4) {
            result = "Fecha de finalización";
        } else if (columnIndex == 5) {
            result = "Fase";
        }
        return result;
    }
}
```

4.1.6.7 ModeloTablaMonitor

```
package com.jabluecatch.util;

import com.jabluecatch.dto.Tarea;
import java.util.Vector;
import javax.swing.table.AbstractTableModel;

/**
 * Clase que representa el modelo para armar las tablas en donde se muestran
 * instancias de las tareas, es decir el monitor de la pantalla principal
 * @author Fausto Almeida
 */
public class ModeloTablaMonitor extends AbstractTableModel {

    Vector<Tarea> filas;

    public ModeloTablaMonitor() {
        filas = new Vector();
    }

    public ModeloTablaMonitor(Vector filas) {
        this.filas = filas;
    }

    public Vector getFilas() {
        return filas;
    }

    public void setFilas(Vector filas) {
        this.filas = filas;
    }

    public int getRowCount() {
        return this.filas.size();
    }

    public int getColumnCount() {
        return 4;
    }

    public Object getValueAt(int rowIndex, int columnIndex) {
        Object result = null;
        Tarea rm = this.filas.get(rowIndex);
        if (columnIndex == 0) {
            result = rm.getReceptor().getNombreBt();
        } else if (columnIndex == 1) {
            result = rm.getArchivoAEnviar().getRuta();
        } else if (columnIndex == 2) {
            result = rm.getEstado();
        } else if (columnIndex == 3) {
            result = rm.getPorcentajeCompletado();
        }

        return result;
    }
```

```
    @Override
    public String getColumnName(int columnIndex) {
        String result = "";
        if (columnIndex == 0) {
            result = "Nombre dispositivo";
        } else if (columnIndex == 1) {
            result = "Archivo a enviar";
        } else if (columnIndex == 2) {
            result = "Estado";
        } else if (columnIndex == 3) {
            result = "Porcentaje completado";
        }
        return result;
    }
}
```

4.1.6.8 ModeloTablaUsuarios

```
package com.jabluecatch.util;

import com.jabluecatch.dto.ConsultaUsuarios;
import java.util.Vector;
import javax.swing.table.AbstractTableModel;

/**
 * Clase que representa el modelo para armar las tablas en donde se muestran
 * instancias de los usuarios
 * @author Fausto Almeida
 */
public class ModeloTablaUsuarios extends AbstractTableModel {

    Vector<ConsultaUsuarios> filas;

    public ModeloTablaUsuarios() {
        filas = new Vector();
    }

    public ModeloTablaUsuarios(Vector filas) {
        this.filas = filas;
    }

    public Vector getFilas() {
        return filas;
    }

    public void setFilas(Vector filas) {
        this.filas = filas;
    }

    public int getRowCount() {
        return this.filas.size();
    }

    public int getColumnCount() {
```

```
        return 5;
    }

    public Object getValueAt(int rowIndex, int columnIndex) {
        ConsultaUsuarios u = this.filas.get(rowIndex);
        Object result = null;
        if (columnIndex == 0) {
            result = u.getIdUsuario();
        } else if (columnIndex == 1) {
            result = u.getUsuario();
        } else if (columnIndex == 2) {
            result = u.getRol();
        } else if (columnIndex == 3) {
            result = u.getNombres();
        } else if (columnIndex == 4) {
            result = u.getApellidos();
        }
        return result;
    }

    @Override
    public String getColumnName(int columnIndex) {
        String result = null;
        if (columnIndex == 0) {
            result = "Id";
        } else if (columnIndex == 1) {
            result = "Usuario";
        } else if (columnIndex == 2) {
            result = "Rol";
        } else if (columnIndex == 3) {
            result = "Nombres";
        } else if (columnIndex == 4) {
            result = "Apellidos";
        }
        return result;
    }
}
```

4.1.6.9 ValidadorNombreDispositivo

```
package com.jabluecatch.util;

/**
 * Clase que permite validar si el nombre de un dispositivo encontrado posee o
 * no caracteres especiales
 * @author Fausto Almeida
 */
public class ValidadorNombreDispositivo {

    public static boolean esValido(String nombreDispositivo) {

        boolean result = true;

        char caracteres[] = nombreDispositivo.toCharArray();
```

```
        for (char c : caracteres){
            if (!(Character.isLetterOrDigit(c) || Character.isSpaceChar(c))) {
                result = false;
            }
        }

        return result;
    }

}
```

4.1.7 Paquete com.jabluecatch.gui.sistema

4.1.7.1 CambiarClave

```
package com.jabluecatch.gui.sistema;

import com.jabluecatch.bo.UsuarioBO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de cambio de clave
 * @author Fausto Almeida
 */
public class CambiarClave extends javax.swing.JDialog {

    /** Creates new form CambiarClave */
    public CambiarClave(PantallaPrincipal padre, boolean modal, String
nombreUsuario) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(nombreUsuario);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
```

```
        lblClave = new javax.swing.JLabel();
        lblReClave = new javax.swing.JLabel();
        txtClave = new javax.swing.JPasswordField();
        txtReClave = new javax.swing.JPasswordField();
        btnGuardar = new javax.swing.JButton();
        btnCancelar = new javax.swing.JButton();
        lblClave1 = new javax.swing.JLabel();
        txtClaveActual = new javax.swing.JPasswordField();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Cambiar clave");
        setBounds(new java.awt.Rectangle(175, 225, 400, 415));
        setResizable(false);

        lblClave.setText("Nueva clave:");

        lblReClave.setText("Reingrese la clave:");

        txtClave.setColumns(12);
        txtClave.setDocument(new com.jabluecatch.util.FormateadorDocumento(15,0));

        txtReClave.setColumns(12);
        txtReClave.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,0));

        btnGuardar.setText("Guardar");
        btnGuardar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGuardarActionPerformed(evt);
            }
        });

        btnCancelar.setText("Cancelar");
        btnCancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCancelarActionPerformed(evt);
            }
        });

        lblClave1.setText("Clave actual:");

        txtClaveActual.setColumns(12);
        txtClaveActual.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,0));

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap(78, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                            .addComponent(lblClave)
                            .addComponent(lblReClave)
                            .addComponent(lblClave1))
                        .addGap(24, 24, 24))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addComponent(btnGuardar)
                        .addGap(23, 23, 23)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(txtReClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtClave, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtClaveActual,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnCancelar))
                .addGap(70, 70, 70))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(28, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtClaveActual,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblClave1))
                .addGap(12, 12, 12)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblClave)
                    .addComponent(txtClave, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblReClave)
                    .addComponent(txtReClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(27, 27, 27)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnGuardar)
                    .addComponent(btnCancelar))
                .addGap(25, 25, 25))
        );

        pack();
    }// </editor-fold>

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        this.dispose();
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {

        int acepta = JOptionPane.showConfirmDialog(this, "Esta a punto de
actualizar su clave. Desea continuar?", "Confirmación", JOptionPane.YES_NO_OPTION);
        //logger.warn(acepta);
        if (acepta == 0) {
            //Validaciones
            if (new String(this.txtClaveActual.getPassword()).trim().equals("")) {
                logger.info("La clave actual no puede ser nula");
                JOptionPane.showMessageDialog(this, "La clave actual no puede ser
nula", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (!new
String(this.txtClaveActual.getPassword()).trim().equals(this.usuario.getClave())) {
                logger.info("La clave actual es incorrecta");
                JOptionPane.showMessageDialog(this, "La clave actual es
incorrecta", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (new String(this.txtClave.getPassword()).trim().equals("")) {
                logger.info("La nueva clave no puede ser nula");
                JOptionPane.showMessageDialog(this, "La nueva clave no puede ser
nula", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (new String(this.txtClave.getPassword()).trim().length() < 4)
{
                logger.info("La nueva clave debe tener al menos 4 caracteres");
                JOptionPane.showMessageDialog(this, "La nueva clave debe tener al
menos 4 caracteres", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (!new String(this.txtClave.getPassword()).equals(new
String(this.txtReClave.getPassword()))) {
                logger.info("Las claves ingresadas no coinciden");
                JOptionPane.showMessageDialog(this, "Las claves ingresadas no
coinciden", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }

            Usuario nuevoUsuario = new Usuario(this.usuario.getIdUsuario(),
                    this.usuario.getIdRol(),
                    this.usuario.getUsuario(),
                    new String(this.txtClave.getPassword()),
                    this.usuario.getNombres(),
                    this.usuario.getApellidos(),
                    this.usuario.getIdentificacion(),
                    this.usuario.getEstado());

            try {
                UsuarioBO.actualizar(nuevoUsuario);
                logger.info("Los datos se han guardado correctamente");
                JOptionPane.showMessageDialog(this, "Los datos se han guardado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
                this.setVisible(false);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
```

```
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
        }
    }

    private void cargaDatos(String nombreUsuario) {
        //Carga usuario
        try {
            this.usuario = UsuarioBO.buscarPorUsuario(nombreUsuario);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError());
            this.setVisible(false);
            return;
        }
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnCancelar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JLabel lblClave;
    private javax.swing.JLabel lblClave1;
    private javax.swing.JLabel lblReClave;
    private javax.swing.JPasswordField txtClave;
    private javax.swing.JPasswordField txtClaveActual;
    private javax.swing.JPasswordField txtReClave;
    // End of variables declaration
    private Usuario usuario;
    private static Logger logger = Logger.getLogger(CambiarClave.class);
}
```

4.1.7.2 ConsultarUsuarios

```
package com.jabluecatch.gui.sistema;

import com.jabluecatch.bo.ConsultaUsuariosBO;
import com.jabluecatch.bo.UsuarioBO;
import com.jabluecatch.dto.ConsultaUsuarios;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.ModeloTablaUsuarios;
import java.util.Vector;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de consulta de usuarios
 * @author Fausto Almeida
 */
public class ConsultarUsuarios extends javax.swing.JDialog {

    /** Creates new form ConsultarUsuarios */
    public ConsultarUsuarios(PantallaPrincipal padre, boolean modal) {
```

```
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblUsuario = new javax.swing.JLabel();
        lblNombre = new javax.swing.JLabel();
        txtUsuario = new javax.swing.JTextField();
        txtNombre = new javax.swing.JTextField();
        btnConsultar = new javax.swing.JButton();
        btnCerrar = new javax.swing.JButton();
        jSeparator1 = new javax.swing.JSeparator();
        jScrollPane1 = new javax.swing.JScrollPane();
        tabConsultaUsuarios = new javax.swing.JTable();
        btnVer = new javax.swing.JButton();
        btnEditar = new javax.swing.JButton();
        btnEliminar = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Consultar usuarios");
        setBounds(new java.awt.Rectangle(175, 225, 600, 400));
        setResizable(false);

        lblUsuario.setText("Usuario:");

        lblNombre.setText("Nombres o Apellidos:");

        txtUsuario.setColumns(12);
        txtUsuario.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,2));
        txtUsuario.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtUsuarioActionPerformed(evt);
            }
        });

        txtNombre.setColumns(20);
        txtNombre.setDocument(new com.jabluecatch.util.FormateadorDocumento(60,3));

        btnConsultar.setText("Consultar");
        btnConsultar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnConsultarActionPerformed(evt);
            }
        });
```

```
        btnCerrar.setText("Cerrar");
        btnCerrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCerrarActionPerformed(evt);
            }
        });

        tabConsultaUsuarios.setModel(new
com.jabluecatch.util.ModeloTablaUsuarios());

tabConsultaUsuarios.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTIO
N);
        jScrollPane1.setViewportView(tabConsultaUsuarios);

        btnVer.setText("Ver");
        btnVer.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnVerActionPerformed(evt);
            }
        });

        btnEditar.setText("Editar");
        btnEditar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnEditarActionPerformed(evt);
            }
        });

        btnEliminar.setText("Eliminar");
        btnEliminar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnEliminarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jSeparator1, javax.swing.GroupLayout.DEFAULT_SIZE, 600,
Short.MAX_VALUE)
            .addGroup(layout.createSequentialGroup()
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(btnVer)
                        .addGap(18, 18, 18)
                        .addComponent(btnEditar)
                        .addGap(18, 18, 18)
                        .addComponent(btnEliminar))
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 562,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(20, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(107, 107, 107)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblUsuario)
                    .addComponent(lblNombre))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(txtNombre,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(btnCerrar)
                    .addComponent(btnConsultar))
                .addContainerGap(112, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(25, 25, 25)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblUsuario)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnConsultar))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblNombre)
                    .addComponent(txtNombre,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnCerrar))
                .addGap(18, 18, 18)
                .addComponent(jSeparator1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
230, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnVer)
                    .addComponent(btnEditar)
                    .addComponent(btnEliminar))
                .addContainerGap(16, Short.MAX_VALUE))
        );

        pack();
```

```
    }// </editor-fold>

    private void txtUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCerrarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }

    private void btnConsultarActionPerformed(java.awt.event.ActionEvent evt) {

        Vector usuarios = null;

        try {
            if (!this.txtUsuario.getText().trim().equals("") &&
this.txtNombre.getText().trim().equals("")) {
                //Solo por usuario
                //logger.warn("Solo por usuario");
                usuarios =
ConsultaUsuariosBO.buscarPorPatronUsuario(this.txtUsuario.getText().trim());

            } else if (this.txtUsuario.getText().trim().equals("") &&
!this.txtNombre.getText().trim().equals("")) {
                //Solo por nombres
                //logger.warn("Solo por nombres");
                usuarios =
ConsultaUsuariosBO.buscarPorPatronNombreYApellido(this.txtNombre.getText().trim());

            } else {
                //Por ambos
                //logger.warn("Por ambos");
                usuarios =
ConsultaUsuariosBO.buscarPorPatronUsuarioYNombreYApellido(this.txtUsuario.getText()
.trim(), this.txtNombre.getText().trim());

            }
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            return;
        }

        this.resultadoConsulta = usuarios;

        this.tabConsultaUsuarios.setModel(new ModeloTablaUsuarios(usuarios));
        this.seteaAnchoColumnas();

    }

    public void ejecutarConsulta() {
        this.btnConsultar.doClick();
    }

    private void btnEditarActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaUsuarios.getSelectedRow() >= 0) {
```

```
            new EditarDatosUsuario(this, true, ((ConsultaUsuarios)
this.resultadoConsulta.get(this.tabConsultaUsuarios.getSelectedRow())).getUsuario()
).setVisible(true);
        }
    }

    private void btnVerActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaUsuarios.getSelectedRow() >= 0) {
            new VerDatosUsuario(this, true, ((ConsultaUsuarios)
this.resultadoConsulta.get(this.tabConsultaUsuarios.getSelectedRow())).getUsuario()
).setVisible(true);
        }
    }

    private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaUsuarios.getSelectedRow() >= 0) {
            String nombreUsuario = ((ConsultaUsuarios)
this.resultadoConsulta.get(this.tabConsultaUsuarios.getSelectedRow())).getUsuario()
;
            if (nombreUsuario.equals("admin")) {
                logger.info("El usuario <admin> no puede ser eliminado");
                JOptionPane.showMessageDialog(this, "El usuario <admin> no puede
ser eliminado", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            int acepta = JOptionPane.showConfirmDialog(this, "Esta a punto de
eliminar al usuario <" + nombreUsuario + ">. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);
            //logger.warn(acepta);
            if (acepta == 0) {
                Usuario usuario = null;
                //Carga usuario y lo inactiva
                try {
                    usuario = UsuarioBO.buscarPorUsuario(nombreUsuario);
                    usuario.setEstado("I");
                    UsuarioBO.actualizar(usuario);
                } catch (ErrorAplicativo ea) {
                    logger.warn(ea.getErrorTecnico());
                    JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                }
                logger.info("El usuario se ha eliminado correctamente");
                JOptionPane.showMessageDialog(this, "El usuario se ha eliminado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
                this.btnConsultar.doClick();
            }
        }
    }

    private void seteaAnchoColumnas() {
        this.tabConsultaUsuarios.getColumn("Id").setPreferredWidth(72);
        this.tabConsultaUsuarios.getColumn("Usuario").setPreferredWidth(120);
        this.tabConsultaUsuarios.getColumn("Rol").setPreferredWidth(120);
        this.tabConsultaUsuarios.getColumn("Nombres").setPreferredWidth(180);
        this.tabConsultaUsuarios.getColumn("Apellidos").setPreferredWidth(180);
    }

    private void cargaDatos() {
```

```
        this.seteaAnchoColumnas();
        this.btnConsultar.doClick();
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnCerrar;
    private javax.swing.JButton btnConsultar;
    private javax.swing.JButton btnEditar;
    private javax.swing.JButton btnEliminar;
    private javax.swing.JButton btnVer;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JLabel lblNombre;
    private javax.swing.JLabel lblUsuario;
    private javax.swing.JTable tabConsultaUsuarios;
    private javax.swing.JTextField txtNombre;
    private javax.swing.JTextField txtUsuario;
    // End of variables declaration
    private Vector resultadoConsulta;
    private static Logger logger = Logger.getLogger(ConsultarUsuarios.class);
}
```

4.1.7.3 CrearUsuario

```
package com.jabluecatch.gui.sistema;

import com.jabluecatch.bo.RolBO;
import com.jabluecatch.bo.UsuarioBO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.Rol;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de creación de usuarios
 * @author Fas
 */
public class CrearUsuario extends javax.swing.JDialog {

    /** Creates new form CrearUsuario */
    public CrearUsuario(PantallaPrincipal padre, boolean modal) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
```

```
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblUsuario = new javax.swing.JLabel();
        lblRol = new javax.swing.JLabel();
        lblClave = new javax.swing.JLabel();
        lblReClave = new javax.swing.JLabel();
        lblNombres = new javax.swing.JLabel();
        lblApellidos = new javax.swing.JLabel();
        lblIdentificacion = new javax.swing.JLabel();
        lblEstado = new javax.swing.JLabel();
        txtUsuario = new javax.swing.JTextField();
        cmbRol = new javax.swing.JComboBox();
        txtClave = new javax.swing.JPasswordField();
        txtReClave = new javax.swing.JPasswordField();
        txtNombres = new javax.swing.JTextField();
        txtApellidos = new javax.swing.JTextField();
        txtIdentificacion = new javax.swing.JTextField();
        cmbEstado = new javax.swing.JComboBox();
        btnGuardar = new javax.swing.JButton();
        btnCancelar = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Crear usuario");
        setBounds(new java.awt.Rectangle(175, 225, 400, 415));
        setResizable(false);

        lblUsuario.setText("Nombre de usuario: ");

        lblRol.setText("Rol:");

        lblClave.setText("Clave:");

        lblReClave.setText("Reingrese la clave:");

        lblNombres.setText("Nombres:");

        lblApellidos.setText("Apellidos:");

        lblIdentificacion.setText("Identificación:");

        lblEstado.setText("Estado:");

        txtUsuario.setColumns(12);
        txtUsuario.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,2));
        txtUsuario.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtUsuarioActionPerformed(evt);
            }
        });

        txtClave.setColumns(12);
        txtClave.setDocument(new com.jabluecatch.util.FormateadorDocumento(15,0));

        txtReClave.setColumns(12);
```

```
        txtReClave.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,0));

        txtNombres.setColumns(20);
        txtNombres.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));

        txtApellidos.setColumns(20);
        txtApellidos.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));

        txtIdentificacion.setColumns(15);
        txtIdentificacion.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,1));

        btnGuardar.setText("Guardar");
        btnGuardar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGuardarActionPerformed(evt);
            }
        });

        btnCancelar.setText("Cancelar");
        btnCancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCancelarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(65, 65, 65)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblClave)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(lblUsuario)
                            .addComponent(lblRol)
                            .addComponent(lblNombres)
                            .addComponent(lblApellidos)
                            .addComponent(lblIdentificacion)
                            .addComponent(lblEstado)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                .addComponent(btnGuardar)
                                .addComponent(lblReClave)))
                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(btnCancelar)
```

```
                            .addComponent(cmbEstado,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtIdentificacion,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtNombres,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtReClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(cmbRol,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(54, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(44, 44, 44)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblUsuario)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblRol)
                    .addComponent(cmbRol, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblClave)
                    .addComponent(txtClave, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblReClave)
                    .addComponent(txtReClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
                    .addComponent(lblNombres)
                    .addComponent(txtNombres,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblApellidos)
                    .addComponent(txtApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblIdentificacion)
                    .addComponent(txtIdentificacion,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblEstado)
                    .addComponent(cmbEstado,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 35,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnGuardar)
                    .addComponent(btnCancelar))
                .addGap(27, 27, 27))
        );

        pack();
    }// </editor-fold>

    private void txtUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        //Validaciones
        if (this.txtUsuario.getText().trim().equals("")) {
            logger.info("El nombre de usuario no puede ser nulo");
            JOptionPane.showMessageDialog(this, "El nombre de usuario no puede ser
nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (new String(this.txtClave.getPassword()).trim().equals("")) {
            logger.info("La clave no puede ser nulo");
            JOptionPane.showMessageDialog(this, "La clave no puede ser nulo",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return;
```

```
        } else if (new String(this.txtClave.getPassword()).trim().length() < 4) {
            logger.info("La clave debe tener al menos 4 caracteres");
            JOptionPane.showMessageDialog(this, "La clave debe tener al menos 4
caracteres", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (!new String(this.txtClave.getPassword()).equals(new
String(this.txtReClave.getPassword()))) {
            logger.info("Las claves ingresadas no coinciden");
            JOptionPane.showMessageDialog(this, "Las claves ingresadas no
coinciden", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.txtNombres.getText().trim().equals("")) {
            logger.info("El nombre no puede ser nulo");
            JOptionPane.showMessageDialog(this, "El nombre no puede ser nulo",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.txtApellidos.getText().trim().equals("")) {
            logger.info("El apellido no puede ser nulo");
            JOptionPane.showMessageDialog(this, "El apellido no puede ser nulo",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.txtIdentificacion.getText().trim().equals("")) {
            logger.info("La identificación no puede ser nula");
            JOptionPane.showMessageDialog(this, "La identificación no puede ser
nula", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        int idRol = (this.cmbRol.getSelectedIndex() == 0 ? 1 : 2);
        String estado = (this.cmbEstado.getSelectedIndex() == 0 ? "A" : "I");
        Usuario nuevoUsuario = new Usuario(idRol,
                this.txtUsuario.getText().trim(),
                new String(this.txtClave.getPassword()),
                this.txtNombres.getText().trim(),
                this.txtApellidos.getText().trim(),
                this.txtIdentificacion.getText().trim(),
                estado);

        try {
            UsuarioBO.insertar(nuevoUsuario);
            logger.info("Los datos se han guardado correctamente");
            JOptionPane.showMessageDialog(this, "Los datos se han guardado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
            this.setVisible(false);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            return;
        }

    }

    private void cargaDatos() {
        //Carga combo de roles
        Vector roles = null;
        try {
            roles = RolBO.buscarPorEstado("A");
            for (int i = 0; i < roles.size(); i++) {
```

```
                this.cmbRol.addItem(((Rol) roles.get(i)).getRol());
            }
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga combo de estados
        this.cmbEstado.addItem("Activo");
        this.cmbEstado.addItem("Inactivo");
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnCancelar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JComboBox cmbEstado;
    private javax.swing.JComboBox cmbRol;
    private javax.swing.JLabel lblApellidos;
    private javax.swing.JLabel lblClave;
    private javax.swing.JLabel lblEstado;
    private javax.swing.JLabel lblIdentificacion;
    private javax.swing.JLabel lblNombres;
    private javax.swing.JLabel lblReClave;
    private javax.swing.JLabel lblRol;
    private javax.swing.JLabel lblUsuario;
    private javax.swing.JTextField txtApellidos;
    private javax.swing.JPasswordField txtClave;
    private javax.swing.JTextField txtIdentificacion;
    private javax.swing.JTextField txtNombres;
    private javax.swing.JPasswordField txtReClave;
    private javax.swing.JTextField txtUsuario;
    // End of variables declaration
    private static Logger logger = Logger.getLogger(CrearUsuario.class);
}
```

4.1.7.4 EditarDatosUsuario

```
package com.jabluecatch.gui.sistema;

import com.jabluecatch.bo.RolBO;
import com.jabluecatch.bo.UsuarioBO;
import com.jabluecatch.entidades.Rol;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de edición de usuarios
 * @author Fas
 */
public class EditarDatosUsuario extends javax.swing.JDialog {
```

```
    /** Creates new form EditarDatosUsuario */
    public EditarDatosUsuario(ConsultarUsuarios padre, boolean modal, String
nombreUsuario) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(padre, nombreUsuario);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblUsuario = new javax.swing.JLabel();
        lblRol = new javax.swing.JLabel();
        lblClave = new javax.swing.JLabel();
        lblReClave = new javax.swing.JLabel();
        lblNombres = new javax.swing.JLabel();
        lblApellidos = new javax.swing.JLabel();
        lblIdentificacion = new javax.swing.JLabel();
        lblEstado = new javax.swing.JLabel();
        txtUsuario = new javax.swing.JTextField();
        cmbRol = new javax.swing.JComboBox();
        txtClave = new javax.swing.JPasswordField();
        txtReClave = new javax.swing.JPasswordField();
        txtNombres = new javax.swing.JTextField();
        txtApellidos = new javax.swing.JTextField();
        txtIdentificacion = new javax.swing.JTextField();
        cmbEstado = new javax.swing.JComboBox();
        btnGuardar = new javax.swing.JButton();
        btnCancelar = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Editar datos usuario");
        setBounds(new java.awt.Rectangle(680, 220, 400, 415));
        setResizable(false);

        lblUsuario.setText("Nombre de usuario: ");

        lblRol.setText("Rol:");

        lblClave.setText("Clave:");

        lblReClave.setText("Reingrese la clave:");

        lblNombres.setText("Nombres:");

        lblApellidos.setText("Apellidos:");

        lblIdentificacion.setText("Identificación:");
```

```
        lblEstado.setText("Estado:");

        txtUsuario.setColumns(12);
        txtUsuario.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,2));
        txtUsuario.setEditable(false);
        txtUsuario.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtUsuarioActionPerformed(evt);
            }
        });

        txtClave.setColumns(12);
        txtClave.setDocument(new com.jabluecatch.util.FormateadorDocumento(15,0));

        txtReClave.setColumns(12);
        txtReClave.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,0));

        txtNombres.setColumns(20);
        txtNombres.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));

        txtApellidos.setColumns(20);
        txtApellidos.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));

        txtIdentificacion.setColumns(15);
        txtIdentificacion.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,1));

        btnGuardar.setText("Guardar");
        btnGuardar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGuardarActionPerformed(evt);
            }
        });

        btnCancelar.setText("Cancelar");
        btnCancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCancelarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(65, 65, 65)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblClave)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                            .addComponent(lblUsuario)
                            .addComponent(lblRol)
                            .addComponent(lblNombres)
                            .addComponent(lblApellidos)
                            .addComponent(lblIdentificacion)
                            .addComponent(lblEstado)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                .addComponent(btnGuardar)
                                .addComponent(lblReClave)))
                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(btnCancelar)
                            .addComponent(cmbEstado,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtIdentificacion,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtNombres,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtReClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(cmbRol,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(54, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(44, 44, 44)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblUsuario)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblRol)
                    .addComponent(cmbRol, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblClave)
                    .addComponent(txtClave, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblReClave)
                    .addComponent(txtReClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblNombres)
                    .addComponent(txtNombres,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblApellidos)
                    .addComponent(txtApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblIdentificacion)
                    .addComponent(txtIdentificacion,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblEstado)
                    .addComponent(cmbEstado,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 35,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnGuardar)
                    .addComponent(btnCancelar))
                .addGap(27, 27, 27))
        );

        pack();
    }// </editor-fold>

    private void txtUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        this.dispose();
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {

        int acepta = JOptionPane.showConfirmDialog(this, "Esta a punto de
actualizar los datos del usuario. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);
        //logger.warn(acepta);
        if (acepta == 0) {
            //Validaciones
            if (this.txtUsuario.getText().trim().equals("")) {
                logger.info("El nombre de usuario no puede ser nulo");
                JOptionPane.showMessageDialog(this, "El nombre de usuario no puede
ser nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (new String(this.txtClave.getPassword()).trim().equals("")) {
                logger.info("La clave no puede ser nulo");
                JOptionPane.showMessageDialog(this, "La clave no puede ser nulo",
"Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (new String(this.txtClave.getPassword()).trim().length() < 4)
{
                logger.info("La clave debe tener al menos 4 caracteres");
                JOptionPane.showMessageDialog(this, "La clave debe tener al menos 4
caracteres", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (!new String(this.txtClave.getPassword()).equals(new
String(this.txtReClave.getPassword()))) {
                logger.info("Las claves ingresadas no coinciden");
                JOptionPane.showMessageDialog(this, "Las claves ingresadas no
coinciden", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.txtNombres.getText().trim().equals("")) {
                logger.info("El nombre no puede ser nulo");
                JOptionPane.showMessageDialog(this, "El nombre no puede ser nulo",
"Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.txtApellidos.getText().trim().equals("")) {
                logger.info("El apellido no puede ser nulo");
                JOptionPane.showMessageDialog(this, "El apellido no puede ser
nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.txtIdentificacion.getText().trim().equals("")) {
                logger.info("La identificación no puede ser nula");
                JOptionPane.showMessageDialog(this, "La identificación no puede ser
nula", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            int idRol = (this.cmbRol.getSelectedIndex() == 0 ? 1 : 2);
            String estado = (this.cmbEstado.getSelectedIndex() == 0 ? "A" : "I");
            Usuario nuevoUsuario = new Usuario(this.usuario.getIdUsuario(),
                    idRol,
                    this.txtUsuario.getText().trim(),
                    new String(this.txtClave.getPassword()),
                    this.txtNombres.getText().trim(),
                    this.txtApellidos.getText().trim(),
                    this.txtIdentificacion.getText().trim(),
```

```
                    estado);

            try {
                UsuarioBO.actualizar(nuevoUsuario);
                logger.info("Los datos se han actualizado correctamente");
                JOptionPane.showMessageDialog(this, "Los datos se han actualizado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
                this.padre.ejecutarConsulta();
                this.setVisible(false);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
        }

    }

    private void cargaDatos(ConsultarUsuarios padre, String nombreUsuario) {
        this.padre = padre;
        //Carga usuario
        try {
            this.usuario = UsuarioBO.buscarPorUsuario(nombreUsuario);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError());
            this.setVisible(false);
            return;
        }
        //Carga combo de roles
        Vector roles = null;
        try {
            roles = RolBO.buscarPorEstado("A");
            for (int i = 0; i < roles.size(); i++) {
                this.cmbRol.addItem(((Rol) roles.get(i)).getRol());
            }
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga combo de estados
        this.cmbEstado.addItem("Activo");
        this.cmbEstado.addItem("Inactivo");
        //Cargar datos del usuario
        this.txtUsuario.setText(this.usuario.getUsuario());
        this.cmbRol.setSelectedIndex(this.usuario.getIdRol() - 1);
        this.txtClave.setText(this.usuario.getClave());
        this.txtReClave.setText(this.usuario.getClave());
        this.txtNombres.setText(this.usuario.getNombres());
        this.txtApellidos.setText(this.usuario.getApellidos());
        this.txtIdentificacion.setText(this.usuario.getIdentificacion());
        this.cmbEstado.setSelectedIndex((this.usuario.getEstado().equals("A") ? 0 :
1));
    }
```

```
    // Variables declaration - do not modify
    private javax.swing.JButton btnCancelar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JComboBox cmbEstado;
    private javax.swing.JComboBox cmbRol;
    private javax.swing.JLabel lblApellidos;
    private javax.swing.JLabel lblClave;
    private javax.swing.JLabel lblEstado;
    private javax.swing.JLabel lblIdentificacion;
    private javax.swing.JLabel lblNombres;
    private javax.swing.JLabel lblReClave;
    private javax.swing.JLabel lblRol;
    private javax.swing.JLabel lblUsuario;
    private javax.swing.JTextField txtApellidos;
    private javax.swing.JPasswordField txtClave;
    private javax.swing.JTextField txtIdentificacion;
    private javax.swing.JTextField txtNombres;
    private javax.swing.JPasswordField txtReClave;
    private javax.swing.JTextField txtUsuario;
    // End of variables declaration
    private Usuario usuario;
    private ConsultarUsuarios padre;
    private static Logger logger = Logger.getLogger(EditarDatosUsuario.class);
}
```

4.1.7.5 ParametrosGenerales

```
package com.jabluecatch.gui.sistema;

import com.jabluecatch.bo.ParametroGeneralBO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.ParametroGeneral;
import com.jabluecatch.util.ErrorAplicativo;
import java.io.File;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de configuración de parametros generales
 * @author Fas
 */
public class ParametrosGenerales extends javax.swing.JDialog {

    /** Creates new form ParametrosGenerales */
    public ParametrosGenerales(PantallaPrincipal padre, boolean modal) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
```

```
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblTiempoBloqueo = new javax.swing.JLabel();
        txtTiempoBloqueo = new javax.swing.JTextField();
        btnGuardar = new javax.swing.JButton();
        btnCerrar = new javax.swing.JButton();
        lblRutaReportes = new javax.swing.JLabel();
        txtRutaReportes = new javax.swing.JTextField();
        btnRutaReportes = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Parametros generales");
        setBounds(new java.awt.Rectangle(175, 225, 350, 150));
        setResizable(false);

        lblTiempoBloqueo.setText("Tiempo de bloqueo (min.): ");

        txtTiempoBloqueo.setColumns(12);
        txtTiempoBloqueo.setDocument(new
com.jabluecatch.util.FormateadorDocumento(4,1));

        btnGuardar.setText("Guardar");
        btnGuardar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGuardarActionPerformed(evt);
            }
        });

        btnCerrar.setText("Cerrar");
        btnCerrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCerrarActionPerformed(evt);
            }
        });

        lblRutaReportes.setText("Ruta de reportes exportados: ");

        txtRutaReportes.setColumns(12);
        txtRutaReportes.setEditable(false);
        txtRutaReportes.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtRutaReportesActionPerformed(evt);
            }
        });

        btnRutaReportes.setText("...");
        btnRutaReportes.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnRutaReportesActionPerformed(evt);
            }
        });
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(101, Short.MAX_VALUE)
                .addComponent(btnGuardar)
                .addGap(18, 18, 18)
                .addComponent(btnCerrar)
                .addGap(155, 155, 155))
            .addGroup(layout.createSequentialGroup()
                .addGap(48, 48, 48)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
false)
                    .addComponent(txtRutaReportes,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(lblTiempoBloqueo)
                            .addComponent(lblRutaReportes))
                        .addGap(18, 18, 18)
                        .addComponent(txtTiempoBloqueo,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(btnRutaReportes)
                .addContainerGap(38, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(44, 44, 44)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblTiempoBloqueo)
                    .addComponent(txtTiempoBloqueo,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
                .addComponent(lblRutaReportes)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtRutaReportes,
javax.swing.GroupLayout.PREFERRED_SIZE, 20, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnRutaReportes))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 35,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
                    .addComponent(btnGuardar)
                    .addComponent(btnCerrar))
                .addGap(23, 23, 23))
        );

        pack();
    }// </editor-fold>

    private void btnCerrarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        //Guardamos el parametro TIEMPO_BLOQUEO
        int lTiempoBloqueo = 0;
        try {
            lTiempoBloqueo = Integer.parseInt(this.txtTiempoBloqueo.getText());
        } catch (NumberFormatException nfe) {
            logger.warn(nfe.getMessage());
            JOptionPane.showMessageDialog(this, "El parametro ingresado no es un
valor numérico válido", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        if (lTiempoBloqueo <= 0) {
            logger.info("El parametro ingresado debe ser mayor que 0");
            JOptionPane.showMessageDialog(this, "El parametro ingresado debe ser
mayor que 0", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        this.tiempoBloqueo.setValor(String.valueOf(lTiempoBloqueo));

        this.rutaReportes.setValor(this.txtRutaReportes.getText().replace("\\",
"\\\\"));

        try {
            //Actualizar el valor de la BD
            ParametroGeneralBO.actualizar(this.tiempoBloqueo);
            ParametroGeneralBO.actualizar(this.rutaReportes);
            logger.info("Los datos se han guardado correctamente");
            JOptionPane.showMessageDialog(this, "Los datos se han guardado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
            this.setVisible(false);

        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
        }

    }

    private void btnRutaReportesActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorRutaReportes.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File ruta = this.selectorRutaReportes.getSelectedFile();
            if (!ruta.isDirectory()) {
                logger.debug("btnRutaReportesActionPerformed - La ruta seleccionada
no es un directorio");
```

```
                JOptionPane.showMessageDialog(this, "La ruta seleccionada no es un
directorio", "Error", JOptionPane.ERROR_MESSAGE);
                return;
            }
            if (!ruta.canWrite()) {
                logger.debug("btnRutaReportesActionPerformed - La ruta seleccionada
no tiene permiso de escritura");
                JOptionPane.showMessageDialog(this, "La ruta seleccionada no tiene
permiso de escritura", "Error", JOptionPane.ERROR_MESSAGE);
                return;
            }
            this.txtRutaReportes.setText(ruta.getAbsolutePath());
        }
    }

    private void txtRutaReportesActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void cargaDatos() {
        try {
            this.tiempoBloqueo =
ParametroGeneralBO.buscarPorParametroYEstado("TIEMPO_BLOQUEO","A");
            this.rutaReportes =
ParametroGeneralBO.buscarPorParametroYEstado("RUTA_REPORTES","A");
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        this.txtTiempoBloqueo.setText(this.tiempoBloqueo.getValor());
        this.txtTiempoBloqueo.setToolTipText(this.tiempoBloqueo.getDescripcion());
        this.txtRutaReportes.setText(this.rutaReportes.getValor());
        this.txtRutaReportes.setToolTipText(this.rutaReportes.getDescripcion());

        this.selectorRutaReportes = new JFileChooser();
        this.selectorRutaReportes.setAcceptAllFileFilterUsed(false);

this.selectorRutaReportes.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);

    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnCerrar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JButton btnRutaReportes;
    private javax.swing.JLabel lblRutaReportes;
    private javax.swing.JLabel lblTiempoBloqueo;
    private javax.swing.JTextField txtRutaReportes;
    private javax.swing.JTextField txtTiempoBloqueo;
    // End of variables declaration
    private ParametroGeneral tiempoBloqueo;
    private ParametroGeneral rutaReportes;
    private JFileChooser selectorRutaReportes;
    private static Logger logger = Logger.getLogger(ParametrosGenerales.class);
}
```

4.1.7.6 VerDatosUsuario

```
package com.jabluecatch.gui.sistema;

import com.jabluecatch.bo.RolBO;
import com.jabluecatch.bo.UsuarioBO;
import com.jabluecatch.entidades.Rol;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.Vector;
import javax.swing.JDialog;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de ver los datos de un usuario
 * @author Fas
 */
public class VerDatosUsuario extends javax.swing.JDialog {

    /** Creates new form VerDatosUsuario */
    public VerDatosUsuario(JDialog padre, boolean modal, String nombreUsuario) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(nombreUsuario);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblUsuario = new javax.swing.JLabel();
        lblRol = new javax.swing.JLabel();
        lblNombres = new javax.swing.JLabel();
        lblApellidos = new javax.swing.JLabel();
        lblIdentificacion = new javax.swing.JLabel();
        lblEstado = new javax.swing.JLabel();
        txtUsuario = new javax.swing.JTextField();
        cmbRol = new javax.swing.JComboBox();
        txtNombres = new javax.swing.JTextField();
        txtApellidos = new javax.swing.JTextField();
        txtIdentificacion = new javax.swing.JTextField();
        cmbEstado = new javax.swing.JComboBox();
        btnAceptar = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Ver datos del usuario");
        setBounds(new java.awt.Rectangle(680, 225, 400, 317));
```

```
        setResizable(false);

        lblUsuario.setText("Nombre de usuario: ");

        lblRol.setText("Rol:");

        lblNombres.setText("Nombres:");

        lblApellidos.setText("Apellidos:");

        lblIdentificacion.setText("Identificación:");

        lblEstado.setText("Estado:");

        txtUsuario.setColumns(12);
        txtUsuario.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,2));
        txtUsuario.setEditable(false);
        txtUsuario.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtUsuarioActionPerformed(evt);
            }
        });

        cmbRol.setEnabled(false);

        txtNombres.setColumns(20);
        txtNombres.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));
        txtNombres.setEditable(false);

        txtApellidos.setColumns(20);
        txtApellidos.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));
        txtApellidos.setEditable(false);

        txtIdentificacion.setColumns(15);
        txtIdentificacion.setDocument(new
com.jabluecatch.util.FormateadorDocumento(20,1));
        txtIdentificacion.setEditable(false);

        cmbEstado.setEnabled(false);

        btnAceptar.setText("Aceptar");
        btnAceptar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAceptarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(65, 65, 65)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblUsuario)
                    .addComponent(lblRol)
                    .addComponent(lblNombres)
                    .addComponent(lblApellidos)
                    .addComponent(lblIdentificacion)
                    .addComponent(lblEstado))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(cmbEstado,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtIdentificacion,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtNombres,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(cmbRol, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(54, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(144, 144, 144)
                .addComponent(btnAceptar, javax.swing.GroupLayout.DEFAULT_SIZE, 80,
Short.MAX_VALUE)
                .addGap(176, 176, 176))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(44, 44, 44)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblUsuario)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cmbRol, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblRol))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtNombres,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblNombres))
```

```
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblApellidos))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtIdentificacion,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblIdentificacion))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cmbEstado,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblEstado))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 29,
Short.MAX_VALUE)
                .addComponent(btnAceptar)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void txtUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }

    private void cargaDatos(String nombreUsuario) {
        //Carga usuario
        try {
            this.usuario = UsuarioBO.buscarPorUsuario(nombreUsuario);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga combo de roles
        Vector roles = null;
        try {
            roles = RolBO.buscarPorEstado("A");
            for (int i = 0; i < roles.size(); i++) {
                this.cmbRol.addItem(((Rol) roles.get(i)).getRol());
            }
        } catch (ErrorAplicativo ea) {
```

```
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga combo de estados
        this.cmbEstado.addItem("Activo");
        this.cmbEstado.addItem("Inactivo");
        //Cargar datos del usuario
        this.txtUsuario.setText(this.usuario.getUsuario());
        this.cmbRol.setSelectedIndex(this.usuario.getIdRol() - 1);
        this.txtNombres.setText(this.usuario.getNombres());
        this.txtApellidos.setText(this.usuario.getApellidos());
        this.txtIdentificacion.setText(this.usuario.getIdentificacion());
        this.cmbEstado.setSelectedIndex((this.usuario.getEstado().equals("A") ? 0 :
1));

    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAceptar;
    private javax.swing.JComboBox cmbEstado;
    private javax.swing.JComboBox cmbRol;
    private javax.swing.JLabel lblApellidos;
    private javax.swing.JLabel lblEstado;
    private javax.swing.JLabel lblIdentificacion;
    private javax.swing.JLabel lblNombres;
    private javax.swing.JLabel lblRol;
    private javax.swing.JLabel lblUsuario;
    private javax.swing.JTextField txtApellidos;
    private javax.swing.JTextField txtIdentificacion;
    private javax.swing.JTextField txtNombres;
    private javax.swing.JTextField txtUsuario;
    // End of variables declaration
    private Usuario usuario;
    private static Logger logger = Logger.getLogger(VerDatosUsuario.class);
}
```

4.1.8 Paquete com.jabluecatch.gui.campania

4.1.8.1 ConsultarCampanias

```
package com.jabluecatch.gui.campania;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.bo.ConsultaCampaniasBO;
import com.jabluecatch.dao.CampaniaDAO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.dto.ConsultaCampanias;
```

```
import com.mydatepicker.main.DatePicker;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.ModeloTablaCampanias;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;
import java.util.Vector;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de consulta de campañas
 * @author Fas
 */
public class ConsultarCampanias extends javax.swing.JDialog {

    /** Creates new form ConsultarCampanias */
    public ConsultarCampanias(PantallaPrincipal padre, boolean modal) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(padre);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblCampania = new javax.swing.JLabel();
        lblFechaReferencial = new javax.swing.JLabel();
        txtCampania = new javax.swing.JTextField();
        btnConsultar = new javax.swing.JButton();
        btnCerrar = new javax.swing.JButton();
        jSeparator1 = new javax.swing.JSeparator();
        jScrollPane1 = new javax.swing.JScrollPane();
        tabConsultaCampanias = new javax.swing.JTable();
        dtfFechaReferencial = new com.jabluecatch.util.DateTextField();
        btnVer = new javax.swing.JButton();
        btnEditar = new javax.swing.JButton();
        btnEliminar = new javax.swing.JButton();
        btnFinalizarCampania = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Consultar campañas");
        setBounds(new java.awt.Rectangle(175, 225, 720, 440));
        setResizable(false);

        lblCampania.setText("Nombre Campaña:");

        lblFechaReferencial.setText("Fecha Referencial:");
```

```
        txtCampania.setColumns(20);
        txtCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));
        txtCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtCampaniaActionPerformed(evt);
            }
        });

        btnConsultar.setText("Consultar");
        btnConsultar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnConsultarActionPerformed(evt);
            }
        });

        btnCerrar.setText("Cerrar");
        btnCerrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCerrarActionPerformed(evt);
            }
        });

        tabConsultaCampanias.setModel(new
com.jabluecatch.util.ModeloTablaCampanias());

tabConsultaCampanias.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTI
ON);
        jScrollPane1.setViewportView(tabConsultaCampanias);

        dtfFechaReferencial.setColumns(10);
        dtfFechaReferencial.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaReferencialMouseReleased(evt);
            }
        });

        btnVer.setText("Ver");
        btnVer.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnVerActionPerformed(evt);
            }
        });

        btnEditar.setText("Editar");
        btnEditar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnEditarActionPerformed(evt);
            }
        });

        btnEliminar.setText("Eliminar");
        btnEliminar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnEliminarActionPerformed(evt);
            }
        });
```

```
        btnFinalizarCampania.setText("Finalizar campaña");
        btnFinalizarCampania.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnFinalizarCampaniaActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jSeparator1, javax.swing.GroupLayout.DEFAULT_SIZE, 720,
Short.MAX_VALUE)
            .addGroup(layout.createSequentialGroup()
                .addGap(163, 163, 163)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblCampania)
                    .addComponent(lblFechaReferencial))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, 180,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(dtfFechaReferencial,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(btnConsultar)
                    .addComponent(btnCerrar))
                .addContainerGap(163, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(btnVer)
                        .addGap(18, 18, 18)
                        .addComponent(btnEditar)
                        .addGap(18, 18, 18)
                        .addComponent(btnEliminar)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(btnFinalizarCampania))
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 682,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(20, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
```

```
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblCampania)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnConsultar))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaReferencial)
                    .addComponent(dtfFechaReferencial,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnCerrar))
                .addGap(18, 18, 18)
                .addComponent(jSeparator1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
268, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnVer)
                    .addComponent(btnEditar)
                    .addComponent(btnEliminar)
                    .addComponent(btnFinalizarCampania))
                .addContainerGap(12, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void txtCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCerrarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }

    private void btnConsultarActionPerformed(java.awt.event.ActionEvent evt) {
        Vector campanias = null;

        try {
            if (!this.txtCampania.getText().trim().equals("") &&
this.dtfFechaReferencial.getText().trim().equals("")) {
                //Solo por campania
                //logger.info("Ejecuta consulta solo por campaña");
                campanias =
ConsultaCampaniasBO.buscarPorPatronCampania(this.txtCampania.getText().trim());
```

```
            } else if (this.txtCampania.getText().trim().equals("") &&
!this.dtfFechaReferencial.getText().trim().equals("")) {
                //Solo por fecha referencial
                //logger.info("Ejecuta consulta solo por fecha referencial");
                campanias =
ConsultaCampaniasBO.buscarPorFechaReferencial(this.dtfFechaReferencial.getText().tr
im());

            } else {
                //Por ambos
                //logger.info("Ejecuta consulta por campaña y fecha referencial");
                campanias =
ConsultaCampaniasBO.buscarPorPatronCampaniaYFechaReferencial(this.txtCampania.getTe
xt().trim(), this.dtfFechaReferencial.getText().trim());

            }
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            return;
        }

        this.resultadoConsulta = campanias;

        this.tabConsultaCampanias.setModel(new ModeloTablaCampanias(campanias));
        this.seteaAnchoColumnas();
    }

    public void ejecutarConsulta() {
        this.btnConsultar.doClick();
    }

    private void btnVerActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaCampanias.getSelectedRow() >= 0) {
            new VerDatosCampania(this, true, ((ConsultaCampanias)
this.resultadoConsulta.get(this.tabConsultaCampanias.getSelectedRow())).getIdCampan
ia()).setVisible(true);
        }
    }

    private void btnEditarActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaCampanias.getSelectedRow() >= 0) {
            //Verificar que la fecha de inicio no haya pasado
            Campania c = null;
            try {
                c = CampaniaDAO.buscarPorPK(((ConsultaCampanias)
this.resultadoConsulta.get(this.tabConsultaCampanias.getSelectedRow())).getIdCampan
ia());
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
            if (c.getFechaInicio().before(Calendar.getInstance().getTime())) {
                logger.info("No se puede editar una campaña cuya fecha de inicio ya
haya pasado");
```

```
                JOptionPane.showMessageDialog(this, "No se puede editar una campaña
cuya fecha de inicio ya haya pasado", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            new EditarDatosCampania(this, true, ((ConsultaCampanias)
this.resultadoConsulta.get(this.tabConsultaCampanias.getSelectedRow())).getIdCampan
ia()).setVisible(true);
        }
    }

    private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaCampanias.getSelectedRow() >= 0) {
            //Verificar que la fecha de inicio no haya pasado
            Campania c = null;
            try {
                c = CampaniaDAO.buscarPorPK(((ConsultaCampanias)
this.resultadoConsulta.get(this.tabConsultaCampanias.getSelectedRow())).getIdCampan
ia());
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
            if (c.getFechaInicio().before(Calendar.getInstance().getTime())) {
                logger.info("No se puede eliminar una campaña cuya fecha de inicio
ya haya pasado");
                JOptionPane.showMessageDialog(this, "No se puede eliminar una
campaña cuya fecha de inicio ya haya pasado", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }

            int acepta = JOptionPane.showConfirmDialog(this, "Esta a punto de
eliminar la campaña <" + c.getCampania() + ">. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);

            if (acepta == 0) {

                try {
                    CampaniaBO.eliminar(c);
                    logger.info("La campaña se ha eliminado correctamente");
                    JOptionPane.showMessageDialog(this, "La campaña se ha eliminado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
                    this.btnConsultar.doClick();
                } catch (ErrorAplicativo ea) {
                    logger.warn(ea.getErrorTecnico());
                    JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                    return;
                }
            }
        }
    }

    private void btnFinalizarCampaniaActionPerformed(java.awt.event.ActionEvent
evt) {
```

```
        if (this.tabConsultaCampanias.getSelectedRow() >= 0) {

            //Cargar campania
            Campania c = null;
            try {
                c = CampaniaDAO.buscarPorPK(((ConsultaCampanias)
this.resultadoConsulta.get(this.tabConsultaCampanias.getSelectedRow())).getIdCampan
ia());
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
            if (c.getFase().equals("F")) {
                logger.info("La campaña ya se encuentra finalizada");
                JOptionPane.showMessageDialog(this, "La campaña ya se encuentra
finalizada", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            if (c.getFechaInicio().after(Calendar.getInstance().getTime())) {
                logger.info("No se puede finalizar una campaña que no haya
iniciado");
                JOptionPane.showMessageDialog(this, "No se puede finalizar una
campaña que no haya iniciado", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }

            int acepta = JOptionPane.showConfirmDialog(this, "Esta a punto de
finalizar la campaña <" + c.getCampania() + ">. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);

            if (acepta == 0) {

                c.setFase("F");
                c.setFechaFin(Calendar.getInstance().getTime());
                try {
                    CampaniaBO.actualizar(c);
                    JOptionPane.showMessageDialog(this, "La campaña se ha
finalizado correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
                    this.btnConsultar.doClick();
                } catch (ErrorAplicativo ea) {
                    logger.warn(ea.getErrorTecnico());
                    JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                    return;
                }
            }
        }
    }

    private void dtfFechaReferencialMouseReleased(java.awt.event.MouseEvent evt) {
        DatePicker dp = new DatePicker(dtfFechaReferencial, new Locale("ES"));
        Date fechaSeleccionada = dp.parseDate(dtfFechaReferencial.getText());
        dp.setSelectedDate(fechaSeleccionada);
        dp.start(dtfFechaReferencial);
    }
```

```
    private void seteaAnchoColumnas() {
        this.tabConsultaCampanias.getColumn("Id").setPreferredWidth(72);
        this.tabConsultaCampanias.getColumn("Campaña").setPreferredWidth(300);
        this.tabConsultaCampanias.getColumn("Tipo").setPreferredWidth(100);
        this.tabConsultaCampanias.getColumn("Fecha de
inicio").setPreferredWidth(160);
        this.tabConsultaCampanias.getColumn("Fecha de
finalización").setPreferredWidth(160);
        this.tabConsultaCampanias.getColumn("Fase").setPreferredWidth(80);
    }

    private void cargaDatos(PantallaPrincipal padre) {
        if (padre.getUsuarioSesion().getIdRol() != 1) {
            this.btnEditar.setEnabled(false);
            this.btnEliminar.setEnabled(false);
        }
        this.seteaAnchoColumnas();
        this.btnConsultar.doClick();
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnCerrar;
    private javax.swing.JButton btnConsultar;
    private javax.swing.JButton btnEditar;
    private javax.swing.JButton btnEliminar;
    private javax.swing.JButton btnFinalizarCampania;
    private javax.swing.JButton btnVer;
    private com.jabluecatch.util.DateTextField dtfFechaReferencial;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblFechaReferencial;
    private javax.swing.JTable tabConsultaCampanias;
    private javax.swing.JTextField txtCampania;
    // End of variables declaration
    private Vector resultadoConsulta;
    private static Logger logger = Logger.getLogger(ConsultarCampanias.class);
}
```

4.1.8.2 CrearCampania

```
package com.jabluecatch.gui.campania;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.gui.PantallaPrincipal;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import com.mydatepicker.main.DatePicker;
import java.io.File;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;
import java.util.Vector;
import javax.swing.JFileChooser;
```

```
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileNameExtensionFilter;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de creación de campañas
 * @author Fas
 */
public class CrearCampania extends javax.swing.JDialog {

    private final int PESO_MAXIMO_KB = 500;

    /** Creates new form CrearCampania */
    public CrearCampania(PantallaPrincipal padre, boolean modal) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jFileChooser1 = new javax.swing.JFileChooser();
        jFileChooser2 = new javax.swing.JFileChooser();
        lblCampania = new javax.swing.JLabel();
        lblDescripcion = new javax.swing.JLabel();
        lblTipo = new javax.swing.JLabel();
        lblFechaInicio = new javax.swing.JLabel();
        lblFechaFin = new javax.swing.JLabel();
        txtCampania = new javax.swing.JTextField();
        btnGuardar = new javax.swing.JButton();
        btnCancelar = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        tarDescripcion = new javax.swing.JTextArea();
        cmbTipo = new javax.swing.JComboBox();
        lblHoraInicio = new javax.swing.JLabel();
        lblHoraFin = new javax.swing.JLabel();
        lblDetalleAudio = new javax.swing.JLabel();
        lblAudio1 = new javax.swing.JLabel();
        txtAudio1 = new javax.swing.JTextField();
        btnAudio1 = new javax.swing.JButton();
        txtAudio2 = new javax.swing.JTextField();
        txtAudio3 = new javax.swing.JTextField();
        lblImagen1 = new javax.swing.JLabel();
        lblAudio2 = new javax.swing.JLabel();
        lblDetalleImagen = new javax.swing.JLabel();
        btnAudio2 = new javax.swing.JButton();
        lblAudio3 = new javax.swing.JLabel();
        btnAudio3 = new javax.swing.JButton();
```

```
        txtImagen1 = new javax.swing.JTextField();
        btnImagen1 = new javax.swing.JButton();
        lblImagen2 = new javax.swing.JLabel();
        btnImagen2 = new javax.swing.JButton();
        btnImagen3 = new javax.swing.JButton();
        txtImagen2 = new javax.swing.JTextField();
        txtImagen3 = new javax.swing.JTextField();
        lblImagen3 = new javax.swing.JLabel();
        dtfFechaInicio = new com.jabluecatch.util.DateTextField();
        dtfFechaFin = new com.jabluecatch.util.DateTextField();
        spiHoraInicio = new javax.swing.JSpinner();
        spiMinutoInicio = new javax.swing.JSpinner();
        spiHoraFin = new javax.swing.JSpinner();
        spiMinutoFin = new javax.swing.JSpinner();
        btnXAudio1 = new javax.swing.JButton();
        btnXAudio2 = new javax.swing.JButton();
        btnXAudio3 = new javax.swing.JButton();
        btnXImagen1 = new javax.swing.JButton();
        btnXImagen2 = new javax.swing.JButton();
        btnXImagen3 = new javax.swing.JButton();
        lblCostoRefMensaje = new javax.swing.JLabel();
        spiCostoRefMensaje = new javax.swing.JSpinner();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Crear campaña");
        setBounds(new java.awt.Rectangle(350, 100, 570, 690));
        setResizable(false);

        lblCampania.setText("Nombre de la campaña: ");

        lblDescripcion.setText("Descripción:");

        lblTipo.setText("Tipo de la campaña:");

        lblFechaInicio.setText("Fecha inicio (dd/mm/yyyy):");

        lblFechaFin.setText("Fecha fin (dd/mm/yyyy):");

        txtCampania.setColumns(30);
        txtCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(180,3));
        txtCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtCampaniaActionPerformed(evt);
            }
        });

        btnGuardar.setText("Guardar");
        btnGuardar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGuardarActionPerformed(evt);
            }
        });

        btnCancelar.setText("Cancelar");
        btnCancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCancelarActionPerformed(evt);
```

```
            }
        });

        tarDescripcion.setColumns(30);
        tarDescripcion.setDocument(new
com.jabluecatch.util.FormateadorDocumento(450,0));
        tarDescripcion.setFont(new java.awt.Font("Tahoma", 0, 11));
        tarDescripcion.setLineWrap(true);
        tarDescripcion.setRows(4);
        tarDescripcion.setWrapStyleWord(true);
        jScrollPane1.setViewportView(tarDescripcion);

        cmbTipo.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                cmbTipoItemStateChanged(evt);
            }
        });
        cmbTipo.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusLost(java.awt.event.FocusEvent evt) {
                cmbTipoFocusLost(evt);
            }
        });
        cmbTipo.addPropertyChangeListener(new java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                cmbTipoPropertyChange(evt);
            }
        });
        cmbTipo.addVetoableChangeListener(new java.beans.VetoableChangeListener() {
            public void vetoableChange(java.beans.PropertyChangeEvent evt)throws
java.beans.PropertyVetoException {
                cmbTipoVetoableChange(evt);
            }
        });

        lblHoraInicio.setText("Hora (hh24 mi):");

        lblHoraFin.setText("Hora (hh24 mi):");

        lblDetalleAudio.setText("Archivos de audio a enviar:");

        lblAudio1.setText("Archivo 1:");

        txtAudio1.setColumns(30);
        txtAudio1.setEditable(false);

        btnAudio1.setText("Examinar");
        btnAudio1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAudio1ActionPerformed(evt);
            }
        });

        txtAudio2.setEditable(false);

        txtAudio3.setEditable(false);

        lblImagen1.setText("Archivo 1:");
```

```
lblAudio2.setText("Archivo 2:");

lblDetalleImagen.setText("Archivos de imagen a enviar:");

btnAudio2.setText("Examinar");
btnAudio2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnAudio2ActionPerformed(evt);
    }
});

lblAudio3.setText("Archivo 3:");

btnAudio3.setText("Examinar");
btnAudio3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnAudio3ActionPerformed(evt);
    }
});

txtImagen1.setColumns(30);
txtImagen1.setEditable(false);

btnImagen1.setText("Examinar");
btnImagen1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnImagen1ActionPerformed(evt);
    }
});

lblImagen2.setText("Archivo 2:");

btnImagen2.setText("Examinar");
btnImagen2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnImagen2ActionPerformed(evt);
    }
});

btnImagen3.setText("Examinar");
btnImagen3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnImagen3ActionPerformed(evt);
    }
});

txtImagen2.setEditable(false);

txtImagen3.setEditable(false);

lblImagen3.setText("Archivo 3:");

dtfFechaInicio.setColumns(10);
dtfFechaInicio.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseReleased(java.awt.event.MouseEvent evt) {
        dtfFechaInicioMouseReleased(evt);
    }
});
```

```
        dtfFechaFin.setColumns(10);
        dtfFechaFin.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaFinMouseReleased(evt);
            }
        });
        dtfFechaFin.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                dtfFechaFinActionPerformed(evt);
            }
        });
        dtfFechaFin.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusLost(java.awt.event.FocusEvent evt) {
                dtfFechaFinFocusLost(evt);
            }
        });

        spiHoraInicio.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23"}));

        spiMinutoInicio.setModel(new javax.swing.SpinnerListModel(new String[]
{"00", "01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12",
"13", "14", "15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26",
"27", "28", "29", "30", "31", "32", "33", "34", "35", "36", "37", "38", "39", "40",
"41", "42", "43", "44", "45", "46", "47", "48", "49", "50", "51", "52", "53", "54",
"55", "56", "57", "58", "59"}));

        spiHoraFin.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", " "}));

        spiMinutoFin.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28",
"29", "30", "31", "32", "33", "34", "35", "36", "37", "38", "39", "40", "41", "42",
"43", "44", "45", "46", "47", "48", "49", "50", "51", "52", "53", "54", "55", "56",
"57", "58", "59", " "}));

        btnXAudio1.setText("X");
        btnXAudio1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXAudio1ActionPerformed(evt);
            }
        });

        btnXAudio2.setText("X");
        btnXAudio2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXAudio2ActionPerformed(evt);
            }
        });

        btnXAudio3.setText("X");
        btnXAudio3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXAudio3ActionPerformed(evt);
```

```
            }
        });

        btnXImagen1.setText("X");
        btnXImagen1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXImagen1ActionPerformed(evt);
            }
        });

        btnXImagen2.setText("X");
        btnXImagen2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXImagen2ActionPerformed(evt);
            }
        });

        btnXImagen3.setText("X");
        btnXImagen3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXImagen3ActionPerformed(evt);
            }
        });

        lblCostoRefMensaje.setText("Costo ref. por mensaje ($):");

        spiCostoRefMensaje.setModel(new
javax.swing.SpinnerNumberModel(Float.valueOf(0.0f), Float.valueOf(0.0f),
Float.valueOf(2.0f), Float.valueOf(0.1f)));

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(lblCostoRefMensaje)
                        .addGap(18, 18, 18)
                        .addComponent(spiCostoRefMensaje,
javax.swing.GroupLayout.PREFERRED_SIZE, 39, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 322,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(lblDetalleImagen)
                                .addGroup(layout.createSequentialGroup()
                                    .addComponent(lblDetalleAudio)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 323,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(lblAudio1)
                                                .addComponent(lblAudio2)
                                                .addComponent(lblAudio3))
                                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(txtAudio2,
javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
292, Short.MAX_VALUE)
                                                .addComponent(txtAudio1,
javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
292, Short.MAX_VALUE)
                                                .addComponent(txtAudio3,
javax.swing.GroupLayout.DEFAULT_SIZE, 292, Short.MAX_VALUE)))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(lblImagen1)
                                                .addComponent(lblImagen2)
                                                .addComponent(lblImagen3))
                                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(txtImagen2,
javax.swing.GroupLayout.DEFAULT_SIZE, 292, Short.MAX_VALUE)
                                                .addComponent(txtImagen1,
javax.swing.GroupLayout.DEFAULT_SIZE, 292, Short.MAX_VALUE)
                                                .addComponent(txtImagen3,
javax.swing.GroupLayout.DEFAULT_SIZE, 292, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                                                    .addComponent(btnGuardar)
                                                    .addGap(75, 75, 75)
                                                    .addComponent(btnCancelar)))))
                                    .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                        .addComponent(btnAudio2)
                                        .addComponent(btnAudio1)
                                        .addComponent(btnAudio3)
                                        .addComponent(btnImagen1)
                                        .addComponent(btnImagen2)
                                        .addComponent(btnImagen3))))
                                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(btnXAudio1)
```

```
                                .addComponent(btnXAudio2)
                                .addComponent(btnXAudio3)
                                .addComponent(btnXImagen1)
                                .addComponent(btnXImagen2)
                                .addComponent(btnXImagen3)))
                            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(lblFechaFin)
                                .addComponent(lblFechaInicio)
                                .addComponent(lblTipo)
                                .addComponent(lblDescripcion)
                                .addComponent(lblCampania))
                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(txtCampania,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE)
                                .addComponent(cmbTipo,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(dtfFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(dtfFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                    .addComponent(lblHoraInicio)
                                    .addComponent(lblHoraFin))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addGroup(layout.createSequentialGroup()
                                        .addComponent(spiHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(spiMinutoInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                    .addGroup(layout.createSequentialGroup()
                                        .addComponent(spiHoraFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(spiMinutoFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
```

```
                                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE))
                            .addGap(77, 77, 77))))
                .addGap(27, 27, 27))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(30, 30, 30)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblCampania))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblDescripcion)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cmbTipo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblTipo))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaInicio)
                    .addComponent(spiHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiMinutoInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(dtfFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblHoraInicio))
                .addGap(9, 9, 9)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaFin)
                    .addComponent(dtfFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiHoraFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiMinutoFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblHoraFin))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblCostoRefMensaje)
                    .addComponent(spiCostoRefMensaje,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 30,
Short.MAX_VALUE)
                .addComponent(lblDetalleAudio)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblAudio1)
                    .addComponent(txtAudio1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnAudio1)
                    .addComponent(btnXAudio1))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnAudio2)
                    .addComponent(txtAudio2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAudio2)
                    .addComponent(btnXAudio2))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnAudio3)
                    .addComponent(txtAudio3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAudio3)
                    .addComponent(btnXAudio3))
                .addGap(33, 33, 33)
                .addComponent(lblDetalleImagen)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnImagen1)
                    .addComponent(txtImagen1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen1)
                    .addComponent(btnXImagen1))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnImagen2)
                    .addComponent(txtImagen2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen2)
                    .addComponent(btnXImagen2))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnImagen3)
                    .addComponent(txtImagen3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen3)
                    .addComponent(btnXImagen3))
                .addGap(29, 29, 29)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnGuardar)
                    .addComponent(btnCancelar))
                .addGap(28, 28, 28))
        );

        pack();
    }// </editor-fold>

    private void txtCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        //Validar nulos
        if (this.txtCampania.getText().trim().equals("")) {
            logger.info("El nombre de la campaña no puede ser nulo");
            JOptionPane.showMessageDialog(this, "El nombre de la campaña no puede
ser nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.tarDescripcion.getText().trim().equals("")) {
            logger.info("La descripcion no puede ser nulo");
            JOptionPane.showMessageDialog(this, "La descripcion no puede ser nulo",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.dtfFechaInicio.getText().trim().equals("")) {
            logger.info("La fecha de inicio no puede ser nula");
            JOptionPane.showMessageDialog(this, "La fecha de inicio no puede ser
nula", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.dtfFechaFin.getText().trim().equals("") &&
this.cmbTipo.getSelectedIndex() == 0) {
            logger.info("La fecha de finalización no puede ser nula a menos que la
campaña sea permanente");
            JOptionPane.showMessageDialog(this, "La fecha de finalización no puede
ser nula a menos que la campaña sea permanente", "Información",
JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.spiHoraFin.getValue().toString().trim().equals("") &&
this.cmbTipo.getSelectedIndex() == 0) {
            logger.info("La hora de finalización no puede ser nula");
            JOptionPane.showMessageDialog(this, "La hora de finalización no puede
ser nula", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
```

```
        } else if (this.spiMinutoFin.getValue().toString().trim().equals("") &&
this.cmbTipo.getSelectedIndex() == 0) {
            logger.info("El minuto de finalización no puede ser nula a menos que la
campaña sea permanente");
            JOptionPane.showMessageDialog(this, "El minuto de finalización no puede
ser nula a menos que la campaña sea permanente", "Información",
JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.txtAudio1.getText().trim().equals("")
                && this.txtAudio2.getText().trim().equals("")
                && this.txtAudio3.getText().trim().equals("")
                && this.txtImagen1.getText().trim().equals("")
                && this.txtImagen2.getText().trim().equals("")
                && this.txtImagen3.getText().trim().equals("")) {
            logger.info("Debe configurar al menos un archivo para su envío");
            JOptionPane.showMessageDialog(this, "Debe configurar al menos un
archivo para su envío", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        if (this.cmbTipo.getSelectedIndex() != 0) {
            int confirma = JOptionPane.showConfirmDialog(this, "Ud ha escogido
crear una campaña permanente, de ser así, la fecha de finalización ingresada no
será tomada en cuenta. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);
            if (confirma != 0) {
                return;
            }
        }
        //Convertir las fechas
        String sFechaInicio = this.dtfFechaInicio.getText() + " " +
this.spiHoraInicio.getValue() + ":" + this.spiMinutoInicio.getValue() + ":00";
        String sFechaFin = this.dtfFechaFin.getText() + " " +
this.spiHoraFin.getValue() + ":" + this.spiMinutoFin.getValue() + ":00";

        Date fechaInicio = null;
        Date fechaFin = null;
        try {
            fechaInicio = FormateadorFecha.aJavaDateTimeDate(sFechaInicio);
            if (this.cmbTipo.getSelectedIndex() == 0) {
                fechaFin = FormateadorFecha.aJavaDateTimeDate(sFechaFin);
            }
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Información",
JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        if (fechaInicio.before(Calendar.getInstance().getTime())) {
            logger.info("La fecha de inicio no puede ser menor a la fecha actual");
            JOptionPane.showMessageDialog(this, "La fecha de inicio no puede ser
menor a la fecha actual", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        if (this.cmbTipo.getSelectedIndex() == 0) {
            if (fechaFin.before(fechaInicio)) {
                logger.info("La fecha de finalización no puede ser menor a la fecha
de inicio");
```

```
                JOptionPane.showMessageDialog(this, "La fecha de finalización no
puede ser menor a la fecha de inicio", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            if (fechaFin.equals(fechaInicio)) {
                logger.info("La fecha de finalización no puede ser igual a la fecha
de inicio");
                JOptionPane.showMessageDialog(this, "La fecha de finalización no
puede ser igual a la fecha de inicio", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }

        //Revisar que los archivos existan
        Vector<DetalleCampania> detallesCampania = new Vector();
        if (!this.txtAudio1.getText().trim().equals("")) {
            detallesCampania.add(new
DetalleCampania(this.txtAudio1.getText().trim().replace("\\", "\\\\"), "A", "A"));
            this.filAudio1 = new File(this.txtAudio1.getText().trim());
            if (!this.filAudio1.exists()) {
                logger.info("El archivo de audio <" + this.filAudio1.getName() + ">
no existe en la tuta especificada");
                JOptionPane.showMessageDialog(this, "El archivo de audio <" +
this.filAudio1.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }
        if (!this.txtAudio2.getText().trim().equals("")) {
            detallesCampania.add(new
DetalleCampania(this.txtAudio2.getText().trim().replace("\\", "\\\\"), "A", "A"));
            this.filAudio2 = new File(this.txtAudio2.getText().trim());
            if (!this.filAudio2.exists()) {
                logger.info("El archivo de audio <" + this.filAudio2.getName() + ">
no existe en la tuta especificada");
                JOptionPane.showMessageDialog(this, "El archivo de audio <" +
this.filAudio2.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }
        if (!this.txtAudio3.getText().trim().equals("")) {
            detallesCampania.add(new
DetalleCampania(this.txtAudio3.getText().trim().replace("\\", "\\\\"), "A", "A"));
            this.filAudio3 = new File(this.txtAudio3.getText().trim());
            if (!this.filAudio3.exists()) {
                logger.info("El archivo de audio <" + this.filAudio3.getName() + ">
no existe en la tuta especificada");
                JOptionPane.showMessageDialog(this, "El archivo de audio <" +
this.filAudio3.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }
        if (!this.txtImagen1.getText().trim().equals("")) {
```

```
            detallesCampania.add(new
DetalleCampania(this.txtImagen1.getText().trim().replace("\\", "\\\\"), "I", "A"));
            this.filImagen1 = new File(this.txtImagen1.getText().trim());
            if (!this.filImagen1.exists()) {
                logger.info("El archivo de imagen <" + this.filImagen1.getName() +
"> no existe en la tuta especificada");
                JOptionPane.showMessageDialog(this, "El archivo de imagen <" +
this.filImagen1.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }
        if (!this.txtImagen2.getText().trim().equals("")) {
            detallesCampania.add(new
DetalleCampania(this.txtImagen2.getText().trim().replace("\\", "\\\\"), "I", "A"));
            this.filImagen2 = new File(this.txtImagen2.getText().trim());
            if (!this.filImagen2.exists()) {
                logger.info("El archivo de imagen <" + this.filImagen2.getName() +
"> no existe en la tuta especificada");
                JOptionPane.showMessageDialog(this, "El archivo de imagen <" +
this.filImagen2.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }
        if (!this.txtImagen3.getText().trim().equals("")) {
            detallesCampania.add(new
DetalleCampania(this.txtImagen3.getText().trim().replace("\\", "\\\\"), "I", "A"));
            this.filImagen3 = new File(this.txtImagen3.getText().trim());
            if (!this.filImagen3.exists()) {
                logger.info("El archivo de imagen <" + this.filImagen3.getName() +
"> no existe en la tuta especificada");
                JOptionPane.showMessageDialog(this, "El archivo de imagen <" +
this.filImagen3.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
        }

        String tipo = this.cmbTipo.getSelectedIndex() == 0 ? "O" : "P";

        Campania nuevaCampania = new Campania(this.txtCampania.getText().trim(),
                this.tarDescripcion.getText().trim(),
                tipo,
                (Float) this.spiCostoRefMensaje.getValue(),
                fechaInicio,
                this.cmbTipo.getSelectedIndex() == 0 ? fechaFin : null,
                "P",
                "A");

        try {
            CampaniaBO.insertar(nuevaCampania, detallesCampania);
            logger.info("Los datos se han guardado correctamente");
            JOptionPane.showMessageDialog(this, "Los datos se han guardado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
            this.setVisible(false);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
```

```
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            return;
        }
    }

    private void btnAudio2ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoAudio.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoAudio.getSelectedFile();

            this.txtAudio2.setText(fichero.getAbsolutePath());
        }
    }

    private boolean archivoExcedePesoMaximo(File fichero) {
        if ((fichero.length() / 1024) > this.PESO_MAXIMO_KB) {
            logger.info("El archivo <" + fichero.getName() + "> excede el peso
máximo permitido (" + this.PESO_MAXIMO_KB + " KB)");
            JOptionPane.showMessageDialog(this, "El archivo <" + fichero.getName()
+ "> excede el peso máximo permitido (" + this.PESO_MAXIMO_KB + " KB)",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return true;
        }
        return false;
    }

    private void btnAudio3ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoAudio.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoAudio.getSelectedFile();
            if (this.archivoExcedePesoMaximo(fichero)) {
                return;
            }
            this.txtAudio3.setText(fichero.getAbsolutePath());
        }
    }

    private void btnAudio1ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoAudio.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoAudio.getSelectedFile();
            if (this.archivoExcedePesoMaximo(fichero)) {
                return;
            }
            this.txtAudio1.setText(fichero.getAbsolutePath());
            logger.debug("Archivo: " + fichero.getAbsolutePath());
        }
    }

    private void btnImagen1ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoImagen.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoImagen.getSelectedFile();
            if (this.archivoExcedePesoMaximo(fichero)) {
                return;
            }
            this.txtImagen1.setText(fichero.getAbsolutePath());
```

```
    }
}

private void btnImagen2ActionPerformed(java.awt.event.ActionEvent evt) {
    int seleccion = this.selectorArchivoImagen.showOpenDialog(this);
    if (seleccion == JFileChooser.APPROVE_OPTION) {
        File fichero = this.selectorArchivoImagen.getSelectedFile();
        if (this.archivoExcedePesoMaximo(fichero)) {
            return;
        }
        this.txtImagen2.setText(fichero.getAbsolutePath());
    }
}

private void btnImagen3ActionPerformed(java.awt.event.ActionEvent evt) {
    int seleccion = this.selectorArchivoImagen.showOpenDialog(this);
    if (seleccion == JFileChooser.APPROVE_OPTION) {
        File fichero = this.selectorArchivoImagen.getSelectedFile();
        if (this.archivoExcedePesoMaximo(fichero)) {
            return;
        }
        this.txtImagen3.setText(fichero.getAbsolutePath());
    }
}

private void btnXAudio1ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtAudio1.setText("");
}

private void btnXAudio2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtAudio2.setText("");
}

private void btnXAudio3ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtAudio3.setText("");
}

private void btnXImagen1ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtImagen1.setText("");
}

private void btnXImagen2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtImagen1.setText("");
}

private void btnXImagen3ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtImagen1.setText("");
}

private void dtfFechaFinFocusLost(java.awt.event.FocusEvent evt) {
}

private void cmbTipoFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
}

private void cmbTipoItemStateChanged(java.awt.event.ItemEvent evt) {
    if (this.cmbTipo.getSelectedIndex() != 0) {
```

```
            this.dtfFechaFin.setEnabled(false);
            this.spiHoraFin.setEnabled(false);
            this.spiMinutoFin.setEnabled(false);
        } else {
            this.dtfFechaFin.setEnabled(true);
            this.spiHoraFin.setEnabled(true);
            this.spiMinutoFin.setEnabled(true);
        }
    }

    private void cmbTipoPropertyChange(java.beans.PropertyChangeEvent evt) {
    }

    private void cmbTipoVetoableChange(java.beans.PropertyChangeEvent evt)throws
java.beans.PropertyVetoException {
    }

    private void dtfFechaInicioMouseReleased(java.awt.event.MouseEvent evt) {
        DatePicker dp = new DatePicker(dtfFechaInicio, new Locale("ES"));
        Date fechaSeleccionada = dp.parseDate(dtfFechaInicio.getText());
        dp.setSelectedDate(fechaSeleccionada);
        dp.start(dtfFechaInicio);
    }

    private void dtfFechaFinActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void dtfFechaFinMouseReleased(java.awt.event.MouseEvent evt) {
        if (this.dtfFechaFin.isEnabled()) {
            DatePicker dp = new DatePicker(dtfFechaFin, new Locale("ES"));
            Date fechaSeleccionada = dp.parseDate(dtfFechaFin.getText());
            dp.setSelectedDate(fechaSeleccionada);
            dp.start(dtfFechaFin);
        }
    }

    private void cargaDatos() {
        //Carga combo de tipos
        this.cmbTipo.addItem("Operativa");
        this.cmbTipo.addItem("Permanente");

this.dtfFechaInicio.setText(FormateadorFecha.aJavaDateString(Calendar.getInstance()
.getTime()));
        this.spiHoraInicio.setValue("08");
        this.spiMinutoInicio.setValue("30");

this.dtfFechaFin.setText(FormateadorFecha.aJavaDateString(Calendar.getInstance().ge
tTime()));
        this.spiHoraFin.setValue("17");
        this.spiMinutoFin.setValue("30");
        this.selectorArchivoAudio = new JFileChooser();
        this.selectorArchivoAudio.setAcceptAllFileFilterUsed(false);
        this.selectorArchivoAudio.setFileFilter(new FileNameExtensionFilter("Audio:
MP3 & WMA", "mp3", "wma"));
        this.selectorArchivoImagen = new JFileChooser();
        this.selectorArchivoImagen.setAcceptAllFileFilterUsed(false);
```

```
        this.selectorArchivoImagen.setFileFilter(new
FileNameExtensionFilter("Imágenes: JPG & GIF", "jpg", "gif"));
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAudio1;
    private javax.swing.JButton btnAudio2;
    private javax.swing.JButton btnAudio3;
    private javax.swing.JButton btnCancelar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JButton btnImagen1;
    private javax.swing.JButton btnImagen2;
    private javax.swing.JButton btnImagen3;
    private javax.swing.JButton btnXAudio1;
    private javax.swing.JButton btnXAudio2;
    private javax.swing.JButton btnXAudio3;
    private javax.swing.JButton btnXImagen1;
    private javax.swing.JButton btnXImagen2;
    private javax.swing.JButton btnXImagen3;
    private javax.swing.JComboBox cmbTipo;
    private com.jabluecatch.util.DateTextField dtfFechaFin;
    private com.jabluecatch.util.DateTextField dtfFechaInicio;
    private javax.swing.JFileChooser jFileChooser1;
    private javax.swing.JFileChooser jFileChooser2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblAudio1;
    private javax.swing.JLabel lblAudio2;
    private javax.swing.JLabel lblAudio3;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblCostoRefMensaje;
    private javax.swing.JLabel lblDescripcion;
    private javax.swing.JLabel lblDetalleAudio;
    private javax.swing.JLabel lblDetalleImagen;
    private javax.swing.JLabel lblFechaFin;
    private javax.swing.JLabel lblFechaInicio;
    private javax.swing.JLabel lblHoraFin;
    private javax.swing.JLabel lblHoraInicio;
    private javax.swing.JLabel lblImagen1;
    private javax.swing.JLabel lblImagen2;
    private javax.swing.JLabel lblImagen3;
    private javax.swing.JLabel lblTipo;
    private javax.swing.JSpinner spiCostoRefMensaje;
    private javax.swing.JSpinner spiHoraFin;
    private javax.swing.JSpinner spiHoraInicio;
    private javax.swing.JSpinner spiMinutoFin;
    private javax.swing.JSpinner spiMinutoInicio;
    private javax.swing.JTextArea tarDescripcion;
    private javax.swing.JTextField txtAudio1;
    private javax.swing.JTextField txtAudio2;
    private javax.swing.JTextField txtAudio3;
    private javax.swing.JTextField txtCampania;
    private javax.swing.JTextField txtImagen1;
    private javax.swing.JTextField txtImagen2;
    private javax.swing.JTextField txtImagen3;
    // End of variables declaration
    private JFileChooser selectorArchivoAudio;
    private JFileChooser selectorArchivoImagen;
    private File filAudio1;
    private File filAudio2;
```

```
    private File filAudio3;
    private File filImagen1;
    private File filImagen2;
    private File filImagen3;
    private static Logger logger = Logger.getLogger(CrearCampania.class);
}
```

4.1.8.3 EditarDatosCampania

```
package com.jabluecatch.gui.campania;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.bo.DetalleCampaniaBO;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.entidades.DetalleCampania;
import com.mydatepicker.main.DatePicker;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import java.io.File;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;
import java.util.Vector;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileNameExtensionFilter;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de edición de campañas
 * @author Fas
 */
public class EditarDatosCampania extends javax.swing.JDialog {

    private final int PESO_MAXIMO_KB = 500;

    /** Creates new form CrearCampania */
    public EditarDatosCampania(ConsultarCampanias padre, boolean modal, int
idCampania) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(padre, idCampania);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
```

```
jFileChooser1 = new javax.swing.JFileChooser();
jFileChooser2 = new javax.swing.JFileChooser();
lblCampania = new javax.swing.JLabel();
lblDescripcion = new javax.swing.JLabel();
lblTipo = new javax.swing.JLabel();
lblFechaInicio = new javax.swing.JLabel();
lblFechaFin = new javax.swing.JLabel();
txtCampania = new javax.swing.JTextField();
btnGuardar = new javax.swing.JButton();
btnCancelar = new javax.swing.JButton();
jScrollPane1 = new javax.swing.JScrollPane();
tarDescripcion = new javax.swing.JTextArea();
cmbTipo = new javax.swing.JComboBox();
lblHoraInicio = new javax.swing.JLabel();
lblHoraFin = new javax.swing.JLabel();
lblDetalleAudio = new javax.swing.JLabel();
lblAudio1 = new javax.swing.JLabel();
txtAudio1 = new javax.swing.JTextField();
btnAudio1 = new javax.swing.JButton();
txtAudio2 = new javax.swing.JTextField();
txtAudio3 = new javax.swing.JTextField();
lblImagen1 = new javax.swing.JLabel();
lblAudio2 = new javax.swing.JLabel();
lblDetalleImagen = new javax.swing.JLabel();
btnAudio2 = new javax.swing.JButton();
lblAudio3 = new javax.swing.JLabel();
btnAudio3 = new javax.swing.JButton();
txtImagen1 = new javax.swing.JTextField();
btnImagen1 = new javax.swing.JButton();
lblImagen2 = new javax.swing.JLabel();
btnImagen2 = new javax.swing.JButton();
btnImagen3 = new javax.swing.JButton();
txtImagen2 = new javax.swing.JTextField();
txtImagen3 = new javax.swing.JTextField();
lblImagen3 = new javax.swing.JLabel();
dtfFechaInicio = new com.jabluecatch.util.DateTextField();
dtfFechaFin = new com.jabluecatch.util.DateTextField();
spiHoraInicio = new javax.swing.JSpinner();
spiMinutoInicio = new javax.swing.JSpinner();
spiHoraFin = new javax.swing.JSpinner();
spiMinutoFin = new javax.swing.JSpinner();
btnXAudio1 = new javax.swing.JButton();
btnXAudio2 = new javax.swing.JButton();
btnXAudio3 = new javax.swing.JButton();
btnXImagen1 = new javax.swing.JButton();
btnXImagen2 = new javax.swing.JButton();
btnXImagen3 = new javax.swing.JButton();
lblCostoRefMensaje = new javax.swing.JLabel();
spiCostoRefMensaje = new javax.swing.JSpinner();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
setTitle("BlueCatch v0.1 - Editar datos campaña");
setBounds(new java.awt.Rectangle(350, 100, 540, 690));
setResizable(false);

lblCampania.setText("Nombre de la campaña: ");
```

```
        lblDescripcion.setText("Descripción:");

        lblTipo.setText("Tipo de la campaña:");

        lblFechaInicio.setText("Fecha inicio (dd/mm/yyyy):");

        lblFechaFin.setText("Fecha fin (dd/mm/yyyy):");

        txtCampania.setColumns(30);
        txtCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(180,3));
        txtCampania.setEditable(false);
        txtCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtCampaniaActionPerformed(evt);
            }
        });

        btnGuardar.setText("Guardar");
        btnGuardar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGuardarActionPerformed(evt);
            }
        });

        btnCancelar.setText("Cancelar");
        btnCancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCancelarActionPerformed(evt);
            }
        });

        tarDescripcion.setColumns(30);
        tarDescripcion.setDocument(new
com.jabluecatch.util.FormateadorDocumento(450,0));
        tarDescripcion.setFont(new java.awt.Font("Tahoma", 0, 11));
        tarDescripcion.setLineWrap(true);
        tarDescripcion.setRows(4);
        tarDescripcion.setWrapStyleWord(true);
        jScrollPane1.setViewportView(tarDescripcion);

        cmbTipo.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                cmbTipoItemStateChanged(evt);
            }
        });

        lblHoraInicio.setText("Hora (hh24 mi):");

        lblHoraFin.setText("Hora (hh24 mi):");

        lblDetalleAudio.setText("Archivos de audio a enviar:");

        lblAudio1.setText("Archivo 1:");

        txtAudio1.setColumns(30);
        txtAudio1.setEditable(false);
```

```
btnAudio1.setText("Examinar");
btnAudio1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnAudio1ActionPerformed(evt);
    }
});

txtAudio2.setEditable(false);

txtAudio3.setEditable(false);

lblImagen1.setText("Archivo 1:");

lblAudio2.setText("Archivo 2:");

lblDetalleImagen.setText("Archivos de imagen a enviar:");

btnAudio2.setText("Examinar");
btnAudio2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnAudio2ActionPerformed(evt);
    }
});

lblAudio3.setText("Archivo 3:");

btnAudio3.setText("Examinar");
btnAudio3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnAudio3ActionPerformed(evt);
    }
});

txtImagen1.setColumns(30);
txtImagen1.setEditable(false);

btnImagen1.setText("Examinar");
btnImagen1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnImagen1ActionPerformed(evt);
    }
});

lblImagen2.setText("Archivo 2:");

btnImagen2.setText("Examinar");
btnImagen2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnImagen2ActionPerformed(evt);
    }
});

btnImagen3.setText("Examinar");
btnImagen3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnImagen3ActionPerformed(evt);
    }
});
```

```
        txtImagen2.setEditable(false);

        txtImagen3.setEditable(false);

        lblImagen3.setText("Archivo 3:");

        dtfFechaInicio.setColumns(10);
        dtfFechaInicio.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaInicioMouseReleased(evt);
            }
        });

        dtfFechaFin.setColumns(10);
        dtfFechaFin.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaFinMouseReleased(evt);
            }
        });
        dtfFechaFin.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusLost(java.awt.event.FocusEvent evt) {
                dtfFechaFinFocusLost(evt);
            }
        });

        spiHoraInicio.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23"}));

        spiMinutoInicio.setModel(new javax.swing.SpinnerListModel(new String[]
{"00", "01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12",
"13", "14", "15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26",
"27", "28", "29", "30", "31", "32", "33", "34", "35", "36", "37", "38", "39", "40",
"41", "42", "43", "44", "45", "46", "47", "48", "49", "50", "51", "52", "53", "54",
"55", "56", "57", "58", "59"}));

        spiHoraFin.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23"}));

        spiMinutoFin.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28",
"29", "30", "31", "32", "33", "34", "35", "36", "37", "38", "39", "40", "41", "42",
"43", "44", "45", "46", "47", "48", "49", "50", "51", "52", "53", "54", "55", "56",
"57", "58", "59"}));

        btnXAudio1.setText("X");
        btnXAudio1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXAudio1ActionPerformed(evt);
            }
        });

        btnXAudio2.setText("X");
        btnXAudio2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                btnXAudio2ActionPerformed(evt);
            }
        });

        btnXAudio3.setText("X");
        btnXAudio3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXAudio3ActionPerformed(evt);
            }
        });

        btnXImagen1.setText("X");
        btnXImagen1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXImagen1ActionPerformed(evt);
            }
        });

        btnXImagen2.setText("X");
        btnXImagen2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXImagen2ActionPerformed(evt);
            }
        });

        btnXImagen3.setText("X");
        btnXImagen3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnXImagen3ActionPerformed(evt);
            }
        });

        lblCostoRefMensaje.setText("Costo ref. por mensaje ($):");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(lblCostoRefMensaje)
                        .addGap(18, 18, 18)
                        .addComponent(spiCostoRefMensaje,
javax.swing.GroupLayout.PREFERRED_SIZE, 39, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 320,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(lblDetalleImagen)
                                .addGroup(layout.createSequentialGroup()
```

```
                                    .addComponent(lblDetalleAudio)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 321,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(lblAudio1)
                                                .addComponent(lblAudio2)
                                                .addComponent(lblAudio3))
                                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(txtAudio2,
javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
290, Short.MAX_VALUE)
                                                .addComponent(txtAudio1,
javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
290, Short.MAX_VALUE)
                                                .addComponent(txtAudio3,
javax.swing.GroupLayout.DEFAULT_SIZE, 290, Short.MAX_VALUE)))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(lblImagen1)
                                                .addComponent(lblImagen2)
                                                .addComponent(lblImagen3))
                                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                                .addComponent(txtImagen2,
javax.swing.GroupLayout.DEFAULT_SIZE, 290, Short.MAX_VALUE)
                                                .addComponent(txtImagen1,
javax.swing.GroupLayout.DEFAULT_SIZE, 290, Short.MAX_VALUE)
                                                .addComponent(txtImagen3,
javax.swing.GroupLayout.DEFAULT_SIZE, 290, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                                                    .addComponent(btnGuardar)
                                                    .addGap(75, 75, 75)
                                                    .addComponent(btnCancelar)))))
                                    .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                        .addComponent(btnAudio2)
                                        .addComponent(btnAudio1)
                                        .addComponent(btnAudio3)
                                        .addComponent(btnImagen1)
                                        .addComponent(btnImagen2)
                                        .addComponent(btnImagen3))))
                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                                .addComponent(btnXAudio1)
                                .addComponent(btnXAudio2)
                                .addComponent(btnXAudio3)
                                .addComponent(btnXImagen1)
                                .addComponent(btnXImagen2)
                                .addComponent(btnXImagen3)))
                            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(lblFechaFin)
                                .addComponent(lblFechaInicio)
                                .addComponent(lblTipo)
                                .addComponent(lblDescripcion)
                                .addComponent(lblCampania))
                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(txtCampania,
javax.swing.GroupLayout.DEFAULT_SIZE, 284, Short.MAX_VALUE)
                                .addComponent(cmbTipo,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(dtfFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(dtfFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                    .addComponent(lblHoraInicio)
                                    .addComponent(lblHoraFin))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addGroup(layout.createSequentialGroup()
                                        .addComponent(spiHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(spiMinutoInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                    .addGroup(layout.createSequentialGroup()
                                        .addComponent(spiHoraFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(spiMinutoFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
```

```
                                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 284, Short.MAX_VALUE))
                            .addGap(77, 77, 77))))
                .addGap(30, 30, 30))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblCampania))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblDescripcion)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cmbTipo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblTipo))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaInicio)
                    .addComponent(spiHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiMinutoInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(dtfFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblHoraInicio))
                .addGap(9, 9, 9)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaFin)
                    .addComponent(dtfFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiHoraFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiMinutoFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblHoraFin))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblCostoRefMensaje)
                    .addComponent(spiCostoRefMensaje,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 34,
Short.MAX_VALUE)
                .addComponent(lblDetalleAudio)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblAudio1)
                    .addComponent(txtAudio1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnAudio1)
                    .addComponent(btnXAudio1))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnAudio2)
                    .addComponent(txtAudio2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAudio2)
                    .addComponent(btnXAudio2))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnAudio3)
                    .addComponent(txtAudio3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAudio3)
                    .addComponent(btnXAudio3))
                .addGap(33, 33, 33)
                .addComponent(lblDetalleImagen)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnImagen1)
                    .addComponent(txtImagen1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen1)
                    .addComponent(btnXImagen1))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnImagen2)
                    .addComponent(txtImagen2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen2)
                    .addComponent(btnXImagen2))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnImagen3)
                    .addComponent(txtImagen3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen3)
                    .addComponent(btnXImagen3))
                .addGap(29, 29, 29)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnGuardar)
                    .addComponent(btnCancelar))
                .addGap(23, 23, 23))
        );

        pack();
    }// </editor-fold>

    private void txtCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        int acepta = JOptionPane.showConfirmDialog(this, "Esta a punto de
actualizar los datos de la campaña. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);
        if (acepta == 0) {
            //Validar nulos
            if (this.txtCampania.getText().trim().equals("")) {
                logger.info("El nombre de la campaña no puede ser nulo");
                JOptionPane.showMessageDialog(this, "El nombre de la campaña no
puede ser nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.tarDescripcion.getText().trim().equals("")) {
                logger.info("La descripcion no puede ser nulo");
                JOptionPane.showMessageDialog(this, "La descripcion no puede ser
nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.dtfFechaInicio.getText().trim().equals("")) {
                logger.info("La fecha de inicio no puede ser nula");
                JOptionPane.showMessageDialog(this, "La fecha de inicio no puede
ser nula", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.dtfFechaFin.getText().trim().equals("") &&
this.cmbTipo.getSelectedIndex() == 0) {
                logger.info("La fecha de finalización no puede ser nula a menos que
la campaña sea permanente");
                JOptionPane.showMessageDialog(this, "La fecha de finalización no
puede ser nula a menos que la campaña sea permanente", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.spiHoraFin.getValue().toString().trim().equals("") &&
this.cmbTipo.getSelectedIndex() == 0) {
```

```
                logger.info("La hora de finalización no puede ser nula");
                JOptionPane.showMessageDialog(this, "La hora de finalización no
puede ser nula", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.spiMinutoFin.getValue().toString().trim().equals("") &&
this.cmbTipo.getSelectedIndex() == 0) {
                logger.info("El minuto de finalización no puede ser nula a menos
que la campaña sea permanente");
                JOptionPane.showMessageDialog(this, "El minuto de finalización no
puede ser nula a menos que la campaña sea permanente", "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            } else if (this.txtAudio1.getText().trim().equals("")
                    && this.txtAudio2.getText().trim().equals("")
                    && this.txtAudio3.getText().trim().equals("")
                    && this.txtImagen1.getText().trim().equals("")
                    && this.txtImagen2.getText().trim().equals("")
                    && this.txtImagen3.getText().trim().equals("")) {
                logger.info("Debe configurar al menos un archivo para su envío");
                JOptionPane.showMessageDialog(this, "Debe configurar al menos un
archivo para su envío", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            if (this.cmbTipo.getSelectedIndex() != 0) {
                int confirma = JOptionPane.showConfirmDialog(this, "Ud ha escogido
actualizar a una campaña permanente, de ser así, la fecha de finalización ingresada
no será tomada en cuenta. Desea continuar?", "Confirmación",
JOptionPane.YES_NO_OPTION);

                if (confirma != 0) {
                    return;
                }
            }
            //Convertir las fechas
            String sFechaInicio = this.dtfFechaInicio.getText() + " " +
this.spiHoraInicio.getValue() + ":" + this.spiMinutoInicio.getValue() + ":00";
            String sFechaFin = this.dtfFechaFin.getText() + " " +
this.spiHoraFin.getValue() + ":" + this.spiMinutoFin.getValue() + ":00";
            Date fechaInicio = null;
            Date fechaFin = null;
            try {
                fechaInicio = FormateadorFecha.aJavaDateTimeDate(sFechaInicio);
                if (this.cmbTipo.getSelectedIndex() == 0) {
                    fechaFin = FormateadorFecha.aJavaDateTimeDate(sFechaFin);
                }
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Información",
JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            if (fechaInicio.before(Calendar.getInstance().getTime())) {
                logger.info("La fecha de inicio no puede ser menor a la fecha
actual");
                JOptionPane.showMessageDialog(this, "La fecha de inicio no puede
ser menor a la fecha actual", "Información", JOptionPane.INFORMATION_MESSAGE);
                return;
            }
```

```
            if (this.cmbTipo.getSelectedIndex() == 0) {
                if (fechaFin.before(fechaInicio)) {
                    logger.info("La fecha de finalización no puede ser menor a la
fecha de inicio");
                    JOptionPane.showMessageDialog(this, "La fecha de finalización
no puede ser menor a la fecha de inicio", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
                if (fechaFin.equals(fechaInicio)) {
                    logger.info("La fecha de finalización no puede ser igual a la
fecha de inicio");
                    JOptionPane.showMessageDialog(this, "La fecha de finalización
no puede ser igual a la fecha de inicio", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
            }

            //Revisar que los archivos existan
            Vector<DetalleCampania> detallesCampania = new Vector();
            if (!this.txtAudio1.getText().trim().equals("")) {
                detallesCampania.add(new
DetalleCampania(this.txtAudio1.getText().trim().replace("\\", "\\\\"), "A", "A"));
                this.filAudio1 = new File(this.txtAudio1.getText().trim());
                if (!this.filAudio1.exists()) {
                    logger.info("El archivo de audio <" + this.filAudio1.getName()
+ "> no existe en la tuta especificada");
                    JOptionPane.showMessageDialog(this, "El archivo de audio <" +
this.filAudio1.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
            }
            if (!this.txtAudio2.getText().trim().equals("")) {
                detallesCampania.add(new
DetalleCampania(this.txtAudio2.getText().trim().replace("\\", "\\\\"), "A", "A"));
                this.filAudio2 = new File(this.txtAudio2.getText().trim());
                if (!this.filAudio2.exists()) {
                    logger.info("El archivo de audio <" + this.filAudio2.getName()
+ "> no existe en la tuta especificada");
                    JOptionPane.showMessageDialog(this, "El archivo de audio <" +
this.filAudio2.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
            }
            if (!this.txtAudio3.getText().trim().equals("")) {
                detallesCampania.add(new
DetalleCampania(this.txtAudio3.getText().trim().replace("\\", "\\\\"), "A", "A"));
                this.filAudio3 = new File(this.txtAudio3.getText().trim());
                if (!this.filAudio3.exists()) {
                    logger.info("El archivo de audio <" + this.filAudio3.getName()
+ "> no existe en la tuta especificada");
                    JOptionPane.showMessageDialog(this, "El archivo de audio <" +
this.filAudio3.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
```

```
                }
            }
            if (!this.txtImagen1.getText().trim().equals("")) {
                detallesCampania.add(new
DetalleCampania(this.txtImagen1.getText().trim().replace("\\", "\\\\"), "I", "A"));
                this.filImagen1 = new File(this.txtImagen1.getText().trim());
                if (!this.filImagen1.exists()) {
                    logger.info("El archivo de imagen <" +
this.filImagen1.getName() + "> no existe en la tuta especificada");
                    JOptionPane.showMessageDialog(this, "El archivo de imagen <" +
this.filImagen1.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
            }
            if (!this.txtImagen2.getText().trim().equals("")) {
                detallesCampania.add(new
DetalleCampania(this.txtImagen2.getText().trim().replace("\\", "\\\\"), "I", "A"));
                this.filImagen2 = new File(this.txtImagen2.getText().trim());
                if (!this.filImagen2.exists()) {
                    logger.info("El archivo de imagen <" +
this.filImagen2.getName() + "> no existe en la tuta especificada");
                    JOptionPane.showMessageDialog(this, "El archivo de imagen <" +
this.filImagen2.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
            }
            if (!this.txtImagen3.getText().trim().equals("")) {
                detallesCampania.add(new
DetalleCampania(this.txtImagen3.getText().trim().replace("\\", "\\\\"), "I", "A"));
                this.filImagen3 = new File(this.txtImagen3.getText().trim());
                if (!this.filImagen3.exists()) {
                    logger.info("El archivo de imagen <" +
this.filImagen3.getName() + "> no existe en la tuta especificada");
                    JOptionPane.showMessageDialog(this, "El archivo de imagen <" +
this.filImagen3.getName() + "> no existe en la tuta especificada", "Información",
JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
            }

            String tipo = this.cmbTipo.getSelectedIndex() == 0 ? "O" : "P";

            Campania nuevaCampania = new Campania(this.campania.getIdCampania(),
                    this.txtCampania.getText().trim(),
                    this.tarDescripcion.getText().trim(),
                    tipo,
                    (Float) this.spiCostoRefMensaje.getValue(),
                    fechaInicio,
                    this.cmbTipo.getSelectedIndex() == 0 ? fechaFin : null,
                    "P",
                    "A");

            try {
                CampaniaBO.actualizar(nuevaCampania, detallesCampania);
                JOptionPane.showMessageDialog(this, "Los datos se han actualizado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
```

```
                this.padre.ejecutarConsulta();
                this.setVisible(false);
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
        }
    }

    private boolean archivoExcedePesoMaximo(File fichero) {
        if ((fichero.length() / 1024) > this.PESO_MAXIMO_KB) {
            logger.info("El archivo <" + fichero.getName() + "> excede el peso
máximo permitido (" + this.PESO_MAXIMO_KB + " KB)");
            JOptionPane.showMessageDialog(this, "El archivo <" + fichero.getName()
+ "> excede el peso máximo permitido (" + this.PESO_MAXIMO_KB + " KB)",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return true;
        }
        return false;
    }

    private void btnAudio2ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoAudio.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoAudio.getSelectedFile();
            if (this.archivoExcedePesoMaximo(fichero)) {
                return;
            }
            this.txtAudio2.setText(fichero.getAbsolutePath());
        }
    }

    private void btnAudio3ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoAudio.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoAudio.getSelectedFile();
            if (this.archivoExcedePesoMaximo(fichero)) {
                return;
            }
            this.txtAudio3.setText(fichero.getAbsolutePath());
        }
    }

    private void btnAudio1ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoAudio.showOpenDialog(this);
        if (seleccion == JFileChooser.APPROVE_OPTION) {
            File fichero = this.selectorArchivoAudio.getSelectedFile();
            if (this.archivoExcedePesoMaximo(fichero)) {
                return;
            }
            this.txtAudio1.setText(fichero.getAbsolutePath());
        }
    }

    private void btnImagen1ActionPerformed(java.awt.event.ActionEvent evt) {
        int seleccion = this.selectorArchivoImagen.showOpenDialog(this);
```

```
    if (seleccion == JFileChooser.APPROVE_OPTION) {
        File fichero = this.selectorArchivoImagen.getSelectedFile();
        if (this.archivoExcedePesoMaximo(fichero)) {
            return;
        }
        this.txtImagen1.setText(fichero.getAbsolutePath());
    }
}

private void btnImagen2ActionPerformed(java.awt.event.ActionEvent evt) {
    int seleccion = this.selectorArchivoImagen.showOpenDialog(this);
    if (seleccion == JFileChooser.APPROVE_OPTION) {
        File fichero = this.selectorArchivoImagen.getSelectedFile();
        if (this.archivoExcedePesoMaximo(fichero)) {
            return;
        }
        this.txtImagen2.setText(fichero.getAbsolutePath());
    }
}

private void btnImagen3ActionPerformed(java.awt.event.ActionEvent evt) {
    int seleccion = this.selectorArchivoImagen.showOpenDialog(this);
    if (seleccion == JFileChooser.APPROVE_OPTION) {
        File fichero = this.selectorArchivoImagen.getSelectedFile();
        if (this.archivoExcedePesoMaximo(fichero)) {
            return;
        }
        this.txtImagen3.setText(fichero.getAbsolutePath());
    }
}

private void btnXAudio1ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtAudio1.setText("");
}

private void btnXAudio2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtAudio2.setText("");
}

private void btnXAudio3ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtAudio3.setText("");
}

private void btnXImagen1ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtImagen1.setText("");
}

private void btnXImagen2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtImagen1.setText("");
}

private void btnXImagen3ActionPerformed(java.awt.event.ActionEvent evt) {
    this.txtImagen1.setText("");
}

private void dtfFechaFinFocusLost(java.awt.event.FocusEvent evt) {
}
```

```
    private void cmbTipoItemStateChanged(java.awt.event.ItemEvent evt) {
        if (this.cmbTipo.getSelectedIndex() != 0) {
            this.dtfFechaFin.setEnabled(false);
            this.spiHoraFin.setEnabled(false);
            this.spiMinutoFin.setEnabled(false);
        } else {
            this.dtfFechaFin.setEnabled(true);
            this.spiHoraFin.setEnabled(true);
            this.spiMinutoFin.setEnabled(true);
        }
    }

    private void dtfFechaInicioMouseReleased(java.awt.event.MouseEvent evt) {
        DatePicker dp = new DatePicker(dtfFechaInicio, new Locale("ES"));
        Date fechaSeleccionada = dp.parseDate(dtfFechaInicio.getText());
        dp.setSelectedDate(fechaSeleccionada);
        dp.start(dtfFechaInicio);
    }

    private void dtfFechaFinMouseReleased(java.awt.event.MouseEvent evt) {
        if (this.dtfFechaFin.isEnabled()) {
            DatePicker dp = new DatePicker(dtfFechaFin, new Locale("ES"));
            Date fechaSeleccionada = dp.parseDate(dtfFechaFin.getText());
            dp.setSelectedDate(fechaSeleccionada);
            dp.start(dtfFechaFin);
        }
    }

    private void cargaDatos(ConsultarCampanias padre, int idCampania) {
        this.padre = padre;
        //Carga campania
        try {
            this.campania = CampaniaBO.buscarPorPK(idCampania);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga detalles
        Vector<DetalleCampania> detalles = new Vector();
        try {
            detalles = DetalleCampaniaBO.buscarPorIdCampania(idCampania);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga combo de tipos
        this.cmbTipo.addItem("Operativa");
        this.cmbTipo.addItem("Permanente");
        //Carga datos de la campania
        this.txtCampania.setText(this.campania.getCampania());
        this.tarDescripcion.setText(this.campania.getDescripcion());
```

```
        this.cmbTipo.setSelectedIndex((this.campania.getTipo().equals("O") ? 0 :
1));


this.dtfFechaInicio.setText(FormateadorFecha.aJavaDateString(this.campania.getFecha
Inicio()));

this.spiHoraInicio.setValue(FormateadorFecha.aJavaHourString(this.campania.getFecha
Inicio()));

this.spiMinutoInicio.setValue(FormateadorFecha.aJavaMinuteString(this.campania.getF
echaInicio()));

        if (this.campania.getFechaFin() != null) {

this.dtfFechaFin.setText(FormateadorFecha.aJavaDateString(this.campania.getFechaFin
()));

this.spiHoraFin.setValue(FormateadorFecha.aJavaHourString(this.campania.getFechaFin
()));

this.spiMinutoFin.setValue(FormateadorFecha.aJavaMinuteString(this.campania.getFech
aFin()));
        } else {
            this.dtfFechaFin.setText("");
            this.spiHoraFin.setValue("00");
            this.spiMinutoFin.setValue("00");
        }

        this.spiCostoRefMensaje.setValue(this.campania.getCostoRefMensaje());

        //Carga cada uno de los detalles
        Vector<String> rutasAudio = new Vector();
        Vector<String> rutasImagen = new Vector();
        for (int i = 0; i < detalles.size(); i++) {
            if (detalles.get(i).getTipo().equals("A")) {
                rutasAudio.add(detalles.get(i).getRuta());
            } else {
                rutasImagen.add(detalles.get(i).getRuta());
            }
        }

        if (rutasAudio.size() > 0) {
            this.txtAudio1.setText(rutasAudio.get(0));
        }
        if (rutasAudio.size() > 1) {
            this.txtAudio2.setText(rutasAudio.get(1));
        }
        if (rutasAudio.size() > 2) {
            this.txtAudio3.setText(rutasAudio.get(2));
        }
        if (rutasImagen.size() > 0) {
            this.txtImagen1.setText(rutasImagen.get(0));
        }
        if (rutasImagen.size() > 1) {
            this.txtImagen1.setText(rutasImagen.get(1));
        }
        if (rutasImagen.size() > 2) {
```

```
            this.txtImagen1.setText(rutasImagen.get(2));
        }

        this.selectorArchivoAudio = new JFileChooser();
        this.selectorArchivoAudio.setAcceptAllFileFilterUsed(false);
        this.selectorArchivoAudio.setFileFilter(new FileNameExtensionFilter("Audio:
MP3 & WMA", "mp3", "wma"));
        this.selectorArchivoImagen = new JFileChooser();
        this.selectorArchivoImagen.setAcceptAllFileFilterUsed(false);
        this.selectorArchivoImagen.setFileFilter(new
FileNameExtensionFilter("Imágenes: JPG & GIF", "jpg", "gif"));
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAudio1;
    private javax.swing.JButton btnAudio2;
    private javax.swing.JButton btnAudio3;
    private javax.swing.JButton btnCancelar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JButton btnImagen1;
    private javax.swing.JButton btnImagen2;
    private javax.swing.JButton btnImagen3;
    private javax.swing.JButton btnXAudio1;
    private javax.swing.JButton btnXAudio2;
    private javax.swing.JButton btnXAudio3;
    private javax.swing.JButton btnXImagen1;
    private javax.swing.JButton btnXImagen2;
    private javax.swing.JButton btnXImagen3;
    private javax.swing.JComboBox cmbTipo;
    private com.jabluecatch.util.DateTextField dtfFechaFin;
    private com.jabluecatch.util.DateTextField dtfFechaInicio;
    private javax.swing.JFileChooser jFileChooser1;
    private javax.swing.JFileChooser jFileChooser2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblAudio1;
    private javax.swing.JLabel lblAudio2;
    private javax.swing.JLabel lblAudio3;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblCostoRefMensaje;
    private javax.swing.JLabel lblDescripcion;
    private javax.swing.JLabel lblDetalleAudio;
    private javax.swing.JLabel lblDetalleImagen;
    private javax.swing.JLabel lblFechaFin;
    private javax.swing.JLabel lblFechaInicio;
    private javax.swing.JLabel lblHoraFin;
    private javax.swing.JLabel lblHoraInicio;
    private javax.swing.JLabel lblImagen1;
    private javax.swing.JLabel lblImagen2;
    private javax.swing.JLabel lblImagen3;
    private javax.swing.JLabel lblTipo;
    private javax.swing.JSpinner spiCostoRefMensaje;
    private javax.swing.JSpinner spiHoraFin;
    private javax.swing.JSpinner spiHoraInicio;
    private javax.swing.JSpinner spiMinutoFin;
    private javax.swing.JSpinner spiMinutoInicio;
    private javax.swing.JTextArea tarDescripcion;
    private javax.swing.JTextField txtAudio1;
    private javax.swing.JTextField txtAudio2;
    private javax.swing.JTextField txtAudio3;
```

```
    private javax.swing.JTextField txtCampania;
    private javax.swing.JTextField txtImagen1;
    private javax.swing.JTextField txtImagen2;
    private javax.swing.JTextField txtImagen3;
    // End of variables declaration
    private ConsultarCampanias padre;
    private Campania campania;
    private JFileChooser selectorArchivoAudio;
    private JFileChooser selectorArchivoImagen;
    private File filAudio1;
    private File filAudio2;
    private File filAudio3;
    private File filImagen1;
    private File filImagen2;
    private File filImagen3;
    private static Logger logger = Logger.getLogger(EditarDatosCampania.class);
}
```

4.1.8.4 VerDatosCampania

```
package com.jabluecatch.gui.campania;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.bo.DetalleCampaniaBO;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import java.util.Vector;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana para ver los datos de una campaña
 * @author Fausto Almeida
 */
public class VerDatosCampania extends javax.swing.JDialog {

    /** Creates new form CrearCampania */
    public VerDatosCampania(JDialog padre, boolean modal, int idCampania) {
        super(padre, modal);
        initComponents();
        cargaDatos(idCampania);
    }

    /** Creates new form CrearCampania */
    public VerDatosCampania(JFrame padre, boolean modal, int idCampania) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(idCampania);
```

```
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jFileChooser1 = new javax.swing.JFileChooser();
    jFileChooser2 = new javax.swing.JFileChooser();
    lblCampania = new javax.swing.JLabel();
    lblDescripcion = new javax.swing.JLabel();
    lblTipo = new javax.swing.JLabel();
    lblFechaInicio = new javax.swing.JLabel();
    lblFechaFin = new javax.swing.JLabel();
    txtCampania = new javax.swing.JTextField();
    btnAceptar = new javax.swing.JButton();
    jScrollPane1 = new javax.swing.JScrollPane();
    tarDescripcion = new javax.swing.JTextArea();
    cmbTipo = new javax.swing.JComboBox();
    lblHoraInicio = new javax.swing.JLabel();
    lblHoraFin = new javax.swing.JLabel();
    lblDetalleAudio = new javax.swing.JLabel();
    lblAudio1 = new javax.swing.JLabel();
    txtAudio1 = new javax.swing.JTextField();
    txtAudio2 = new javax.swing.JTextField();
    txtAudio3 = new javax.swing.JTextField();
    lblImagen1 = new javax.swing.JLabel();
    lblAudio2 = new javax.swing.JLabel();
    lblDetalleImagen = new javax.swing.JLabel();
    lblAudio3 = new javax.swing.JLabel();
    txtImagen1 = new javax.swing.JTextField();
    lblImagen2 = new javax.swing.JLabel();
    txtImagen2 = new javax.swing.JTextField();
    txtImagen3 = new javax.swing.JTextField();
    lblImagen3 = new javax.swing.JLabel();
    ftxFechaInicio = new javax.swing.JFormattedTextField();
    ftxFechaFin = new javax.swing.JFormattedTextField();
    spiHoraInicio = new javax.swing.JSpinner();
    spiMinutoInicio = new javax.swing.JSpinner();
    spiHoraFin = new javax.swing.JSpinner();
    spiMinutoFin = new javax.swing.JSpinner();
    spiCostoRefMensaje = new javax.swing.JSpinner();
    lblCostoRefMensaje = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
    setTitle("BlueCatch v0.1 - Ver datos campaña");
    setBounds(new java.awt.Rectangle(350, 100, 500, 670));
    setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
    setResizable(false);

    lblCampania.setText("Nombre de la campaña: ");

    lblDescripcion.setText("Descripción:");
```

```
        lblTipo.setText("Tipo de la campaña:");

        lblFechaInicio.setText("Fecha inicio (dd/mm/yyyy):");

        lblFechaFin.setText("Fecha fin (dd/mm/yyyy):");

        txtCampania.setColumns(30);
        txtCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(180,3));
        txtCampania.setEditable(false);
        txtCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtCampaniaActionPerformed(evt);
            }
        });

        btnAceptar.setText("Aceptar");
        btnAceptar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAceptarActionPerformed(evt);
            }
        });

        jScrollPane1.setEnabled(false);

        tarDescripcion.setColumns(30);
        tarDescripcion.setDocument(new
com.jabluecatch.util.FormateadorDocumento(450,0));
        tarDescripcion.setEditable(false);
        tarDescripcion.setFont(new java.awt.Font("Tahoma", 0, 11));
        tarDescripcion.setLineWrap(true);
        tarDescripcion.setRows(4);
        tarDescripcion.setWrapStyleWord(true);
        jScrollPane1.setViewportView(tarDescripcion);

        cmbTipo.setEnabled(false);

        lblHoraInicio.setText("Hora (hh24 mi):");

        lblHoraFin.setText("Hora (hh24 mi):");

        lblDetalleAudio.setText("Archivos de audio a enviar:");

        lblAudio1.setText("Archivo 1:");

        txtAudio1.setColumns(50);
        txtAudio1.setEditable(false);

        txtAudio2.setEditable(false);

        txtAudio3.setEditable(false);

        lblImagen1.setText("Archivo 1:");

        lblAudio2.setText("Archivo 2:");

        lblDetalleImagen.setText("Archivos de imagen a enviar:");
```

```
        lblAudio3.setText("Archivo 3:");

        txtImagen1.setColumns(30);
        txtImagen1.setEditable(false);

        lblImagen2.setText("Archivo 2:");

        txtImagen2.setEditable(false);

        txtImagen3.setEditable(false);

        lblImagen3.setText("Archivo 3:");

        ftxFechaInicio.setColumns(10);
        ftxFechaInicio.setEditable(false);
        ftxFechaInicio.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(new
javax.swing.text.DateFormatter(java.text.DateFormat.getDateInstance(java.text.DateF
ormat.MEDIUM))));

        ftxFechaFin.setColumns(10);
        ftxFechaFin.setEditable(false);
        ftxFechaFin.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(new
javax.swing.text.DateFormatter(java.text.DateFormat.getDateInstance(java.text.DateF
ormat.MEDIUM))));

        spiHoraInicio.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23"}));
        spiHoraInicio.setEnabled(false);

        spiMinutoInicio.setModel(new javax.swing.SpinnerListModel(new String[]
{"00", "01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12",
"13", "14", "15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26",
"27", "28", "29", "30", "31", "32", "33", "34", "35", "36", "37", "38", "39", "40",
"41", "42", "43", "44", "45", "46", "47", "48", "49", "50", "51", "52", "53", "54",
"55", "56", "57", "58", "59"}));
        spiMinutoInicio.setEnabled(false);

        spiHoraFin.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23"}));
        spiHoraFin.setEnabled(false);

        spiMinutoFin.setModel(new javax.swing.SpinnerListModel(new String[] {"00",
"01", "02", "03", "04", "05", "06", "07", "08", "09", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28",
"29", "30", "31", "32", "33", "34", "35", "36", "37", "38", "39", "40", "41", "42",
"43", "44", "45", "46", "47", "48", "49", "50", "51", "52", "53", "54", "55", "56",
"57", "58", "59"}));
        spiMinutoFin.setEnabled(false);

        spiCostoRefMensaje.setEnabled(false);

        lblCostoRefMensaje.setText("Costo ref. por mensaje ($):");
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(210, 210, 210)
                .addComponent(btnAceptar)
                .addContainerGap(219, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(lblCostoRefMensaje)
                        .addGap(18, 18, 18)
                        .addComponent(spiCostoRefMensaje,
javax.swing.GroupLayout.PREFERRED_SIZE, 39, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addContainerGap())

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(lblFechaFin)
                                .addComponent(lblFechaInicio)
                                .addComponent(lblTipo)
                                .addComponent(lblDescripcion)
                                .addComponent(lblCampania))
                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                                .addComponent(cmbTipo,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jScrollPane1)
                                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                        .addComponent(ftxFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                        .addComponent(ftxFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                    .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                        .addComponent(lblHoraInicio)
                                        .addComponent(lblHoraFin))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                        .addGroup(layout.createSequentialGroup()
```

```
                                            .addComponent(spiHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                            .addComponent(spiMinutoInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                        .addGroup(layout.createSequentialGroup()
                                            .addComponent(spiHoraFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                            .addComponent(spiMinutoFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                                .addComponent(txtCampania))
                            .addContainerGap(37, Short.MAX_VALUE))
                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(lblDetalleImagen)
                                .addGroup(layout.createSequentialGroup()
                                    .addComponent(lblDetalleAudio)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 301,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                            .addComponent(lblAudio1)
                                            .addComponent(lblAudio2)
                                            .addComponent(lblAudio3))
                                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                            .addComponent(txtAudio2,
javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
365, Short.MAX_VALUE)
                                            .addComponent(txtAudio3,
javax.swing.GroupLayout.DEFAULT_SIZE, 365, Short.MAX_VALUE)
                                            .addComponent(txtAudio1,
javax.swing.GroupLayout.DEFAULT_SIZE, 365, Short.MAX_VALUE)))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                            .addComponent(lblImagen1)
                                            .addComponent(lblImagen2)
                                            .addComponent(lblImagen3))
                                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                                            .addComponent(txtImagen2,
javax.swing.GroupLayout.DEFAULT_SIZE, 365, Short.MAX_VALUE)
                                            .addComponent(txtImagen1,
javax.swing.GroupLayout.DEFAULT_SIZE, 365, Short.MAX_VALUE)
                                            .addComponent(txtImagen3,
javax.swing.GroupLayout.DEFAULT_SIZE, 365, Short.MAX_VALUE)))))
                            .addGap(37, 37, 37)))))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(26, 26, 26)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblCampania))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblDescripcion)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cmbTipo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblTipo))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaInicio)
                    .addComponent(spiHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiMinutoInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(ftxFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblHoraInicio))
                .addGap(9, 9, 9)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaFin)
                    .addComponent(ftxFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiHoraFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spiMinutoFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addComponent(lblHoraFin))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblCostoRefMensaje)
                    .addComponent(spiCostoRefMensaje,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(21, 21, 21)
                .addComponent(lblDetalleAudio)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblAudio1)
                    .addComponent(txtAudio1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtAudio2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAudio2))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtAudio3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAudio3))
                .addGap(33, 33, 33)
                .addComponent(lblDetalleImagen)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtImagen1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen1))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtImagen2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen2))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtImagen3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblImagen3))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 30,
Short.MAX_VALUE)
                .addComponent(btnAceptar)
```

```
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void txtCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }

    private void cargaDatos(int idCampania) {
        //Carga campania
        try {
            this.campania = CampaniaBO.buscarPorPK(idCampania);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga detalles
        Vector<DetalleCampania> detalles = new Vector();
        try {
            detalles = DetalleCampaniaBO.buscarPorIdCampania(idCampania);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.setVisible(false);
            return;
        }
        //Carga combo de tipos
        this.cmbTipo.addItem("Operativa");
        this.cmbTipo.addItem("Permanente");
        //Carga datos de la campania
        this.txtCampania.setText(this.campania.getCampania());
        this.tarDescripcion.setText(this.campania.getDescripcion());
        this.cmbTipo.setSelectedIndex((this.campania.getTipo().equals("O") ? 0 :
1));


this.ftxFechaInicio.setText(FormateadorFecha.aJavaDateString(this.campania.getFecha
Inicio()));

this.spiHoraInicio.setValue(FormateadorFecha.aJavaHourString(this.campania.getFecha
Inicio()));

this.spiMinutoInicio.setValue(FormateadorFecha.aJavaMinuteString(this.campania.getF
echaInicio()));
        if (this.campania.getFechaFin() != null) {

this.ftxFechaFin.setText(FormateadorFecha.aJavaDateString(this.campania.getFechaFin
()));
```

```
this.spiHoraFin.setValue(FormateadorFecha.aJavaHourString(this.campania.getFechaFin
()));

this.spiMinutoFin.setValue(FormateadorFecha.aJavaMinuteString(this.campania.getFech
aFin()));
        } else {
            this.ftxFechaFin.setText("");
            this.spiHoraFin.setValue("00");
            this.spiMinutoFin.setValue("00");
        }

        this.spiCostoRefMensaje.setValue(this.campania.getCostoRefMensaje());

        Vector<String> rutasAudio = new Vector();
        Vector<String> rutasImagen = new Vector();
        for (int i = 0; i < detalles.size(); i++) {
            if (detalles.get(i).getTipo().equals("A")) {
                rutasAudio.add(detalles.get(i).getRuta());
            } else {
                rutasImagen.add(detalles.get(i).getRuta());
            }
        }

        if (rutasAudio.size() > 0) {
            this.txtAudio1.setText(rutasAudio.get(0));
        }
        if (rutasAudio.size() > 1) {
            this.txtAudio2.setText(rutasAudio.get(1));
        }
        if (rutasAudio.size() > 2) {
            this.txtAudio3.setText(rutasAudio.get(2));
        }
        if (rutasImagen.size() > 0) {
            this.txtImagen1.setText(rutasImagen.get(0));
        }
        if (rutasImagen.size() > 1) {
            this.txtImagen1.setText(rutasImagen.get(1));
        }
        if (rutasImagen.size() > 2) {
            this.txtImagen1.setText(rutasImagen.get(2));
        }

    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAceptar;
    private javax.swing.JComboBox cmbTipo;
    private javax.swing.JFormattedTextField ftxFechaFin;
    private javax.swing.JFormattedTextField ftxFechaInicio;
    private javax.swing.JFileChooser jFileChooser1;
    private javax.swing.JFileChooser jFileChooser2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblAudio1;
    private javax.swing.JLabel lblAudio2;
    private javax.swing.JLabel lblAudio3;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblCostoRefMensaje;
    private javax.swing.JLabel lblDescripcion;
```

```
    private javax.swing.JLabel lblDetalleAudio;
    private javax.swing.JLabel lblDetalleImagen;
    private javax.swing.JLabel lblFechaFin;
    private javax.swing.JLabel lblFechaInicio;
    private javax.swing.JLabel lblHoraFin;
    private javax.swing.JLabel lblHoraInicio;
    private javax.swing.JLabel lblImagen1;
    private javax.swing.JLabel lblImagen2;
    private javax.swing.JLabel lblImagen3;
    private javax.swing.JLabel lblTipo;
    private javax.swing.JSpinner spiCostoRefMensaje;
    private javax.swing.JSpinner spiHoraFin;
    private javax.swing.JSpinner spiHoraInicio;
    private javax.swing.JSpinner spiMinutoFin;
    private javax.swing.JSpinner spiMinutoInicio;
    private javax.swing.JTextArea tarDescripcion;
    private javax.swing.JTextField txtAudio1;
    private javax.swing.JTextField txtAudio2;
    private javax.swing.JTextField txtAudio3;
    private javax.swing.JTextField txtCampania;
    private javax.swing.JTextField txtImagen1;
    private javax.swing.JTextField txtImagen2;
    private javax.swing.JTextField txtImagen3;
    // End of variables declaration
    private Campania campania;
    private static Logger logger = Logger.getLogger(VerDatosCampania.class);
}
```

4.1.9 Paquete com.jabluecatch.gui.reportes

4.1.9.1 CampaniaParametrizable

```
package com.jabluecatch.gui.reportes;

import com.jabluecatch.dto.ConsultaCampanias;

/**
 * Interface necesaria para reutilizar la lista de valores (LOV) de campañas.
 * @author Fausto Almeida
 */
public interface CampaniaParametrizable {

    public void setCampania(ConsultaCampanias campania);
}
```

4.1.9.2 LOVCampanias

```
package com.jabluecatch.gui.reportes;

import com.jabluecatch.bo.ConsultaCampaniasBO;
import com.jabluecatch.dto.ConsultaCampanias;
import com.mydatepicker.main.DatePicker;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.ModeloTablaCampanias;
import java.util.Date;
import java.util.Locale;
import java.util.Vector;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de lista de valores (LOV) de selección
 * de campañas
 * @author Fas
 */
public class LOVCampanias extends javax.swing.JDialog {

    /** Creates new form ConsultarCampanias */
    public LOVCampanias(JFrame padre, boolean modal) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos(padre);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblCampania = new javax.swing.JLabel();
        lblFechaReferencial = new javax.swing.JLabel();
        txtCampania = new javax.swing.JTextField();
        btnConsultar = new javax.swing.JButton();
        btnCerrar = new javax.swing.JButton();
        jSeparator1 = new javax.swing.JSeparator();
        jScrollPane1 = new javax.swing.JScrollPane();
        tabConsultaCampanias = new javax.swing.JTable();
        dtfFechaReferencial = new com.jabluecatch.util.DateTextField();
        btnSeleccionar = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Lista de campañas");
        setBounds(new java.awt.Rectangle(175, 225, 720, 440));
        setResizable(false);

        lblCampania.setText("Nombre Campaña:");
```

```
        lblFechaReferencial.setText("Fecha Referencial:");

        txtCampania.setColumns(20);
        txtCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(60,3));
        txtCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtCampaniaActionPerformed(evt);
            }
        });

        btnConsultar.setText("Consultar");
        btnConsultar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnConsultarActionPerformed(evt);
            }
        });

        btnCerrar.setText("Cerrar");
        btnCerrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCerrarActionPerformed(evt);
            }
        });

        tabConsultaCampanias.setModel(new
com.jabluecatch.util.ModeloTablaCampanias());

tabConsultaCampanias.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTI
ON);
        jScrollPane1.setViewportView(tabConsultaCampanias);

        dtfFechaReferencial.setColumns(10);
        dtfFechaReferencial.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaReferencialMouseReleased(evt);
            }
        });

        btnSeleccionar.setText("Seleccionar");
        btnSeleccionar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSeleccionarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jSeparator1, javax.swing.GroupLayout.DEFAULT_SIZE, 720,
Short.MAX_VALUE)
            .addGroup(layout.createSequentialGroup()
                .addGap(163, 163, 163)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblCampania)
```

```
                    .addComponent(lblFechaReferencial))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, 180,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(dtfFechaReferencial,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(btnConsultar)
                    .addComponent(btnCerrar))
                .addContainerGap(163, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(btnSeleccionar)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 682,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(20, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(31, 31, 31)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblCampania)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnConsultar))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaReferencial)
                    .addComponent(dtfFechaReferencial,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnCerrar))
                .addGap(18, 18, 18)
                .addComponent(jSeparator1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
268, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(btnSeleccionar)
                .addContainerGap(12, Short.MAX_VALUE))
        );
```

```
        pack();
    }// </editor-fold>

    private void txtCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnCerrarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }

    private void btnConsultarActionPerformed(java.awt.event.ActionEvent evt) {
        Vector campanias = null;

        try {
            if (!this.txtCampania.getText().trim().equals("") &&
this.dtfFechaReferencial.getText().trim().equals("")) {
                //Solo por campania
                logger.info("Ejecuta consulta solo por usuario");
                campanias =
ConsultaCampaniasBO.buscarPorPatronCampania(this.txtCampania.getText().trim());

            } else if (this.txtCampania.getText().trim().equals("") &&
!this.dtfFechaReferencial.getText().trim().equals("")) {
                //Solo por fecha referencial
                logger.info("Ejecuta consulta solo por nombre");
                campanias =
ConsultaCampaniasBO.buscarPorFechaReferencial(this.dtfFechaReferencial.getText().tr
im());

            } else {
                //Por ambos
                logger.info("Ejecuta consulta por usuario y nombre");
                campanias =
ConsultaCampaniasBO.buscarPorPatronCampaniaYFechaReferencial(this.txtCampania.getTe
xt().trim(), this.dtfFechaReferencial.getText().trim());

            }
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            return;
        }

        this.resultadoConsulta = campanias;

        this.tabConsultaCampanias.setModel(new ModeloTablaCampanias(campanias));
        this.seteaAnchoColumnas();
    }

    public void ejecutarConsulta() {
        this.btnConsultar.doClick();
    }

    private void btnSeleccionarActionPerformed(java.awt.event.ActionEvent evt) {
        if (this.tabConsultaCampanias.getSelectedRow() >= 0) {
```

```
            CampaniaParametrizable cs = (CampaniaParametrizable) this.padre;
            cs.setCampania(((ConsultaCampanias)
this.resultadoConsulta.get(this.tabConsultaCampanias.getSelectedRow())));
            this.dispose();
        }
    }

    private void dtfFechaReferencialMouseReleased(java.awt.event.MouseEvent evt) {
        DatePicker dp = new DatePicker(dtfFechaReferencial, new Locale("ES"));
        Date fechaSeleccionada = dp.parseDate(dtfFechaReferencial.getText());
        dp.setSelectedDate(fechaSeleccionada);
        dp.start(dtfFechaReferencial);
    }

    private void seteaAnchoColumnas() {
        this.tabConsultaCampanias.getColumn("Id").setPreferredWidth(72);
        this.tabConsultaCampanias.getColumn("Campaña").setPreferredWidth(300);
        this.tabConsultaCampanias.getColumn("Tipo").setPreferredWidth(100);
        this.tabConsultaCampanias.getColumn("Fecha de
inicio").setPreferredWidth(160);
        this.tabConsultaCampanias.getColumn("Fecha de
finalización").setPreferredWidth(160);
        this.tabConsultaCampanias.getColumn("Fase").setPreferredWidth(80);
    }

    private void cargaDatos(JFrame padre) {
        this.padre = padre;
        this.seteaAnchoColumnas();
        this.btnConsultar.doClick();
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnCerrar;
    private javax.swing.JButton btnConsultar;
    private javax.swing.JButton btnSeleccionar;
    private com.jabluecatch.util.DateTextField dtfFechaReferencial;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblFechaReferencial;
    private javax.swing.JTable tabConsultaCampanias;
    private javax.swing.JTextField txtCampania;
    // End of variables declaration
    private JFrame padre;
    private Vector resultadoConsulta;
    private static Logger logger = Logger.getLogger(LOVCampanias.class);
}
```

4.1.9.3 ReporteBitacoraCampania

```
package com.jabluecatch.gui.reportes;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.bo.GestorReportesBO;
import com.jabluecatch.dto.ConsultaCampanias;
import com.jabluecatch.entidades.Campania;
```

```
import com.jabluecatch.util.ErrorAplicativo;
import java.util.HashMap;
import java.util.Map;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de ejecución del reporte de bitácora de
 * campaña
 * @author Fas
 */
public class ReporteBitacoraCampania extends javax.swing.JFrame implements
CampaniaParametrizable {

    /** Creates new form ReporteUsuarios */
    public ReporteBitacoraCampania() {
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnGenerarReporte = new javax.swing.JButton();
        lblIdCampania = new javax.swing.JLabel();
        txtIdCampania = new javax.swing.JTextField();
        btnLOVCampania = new javax.swing.JButton();
        txtCampania = new javax.swing.JTextField();
        lblCampania = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Reporte de Bitácora de Campaña");
        setBounds(new java.awt.Rectangle(350, 100, 370, 150));

        btnGenerarReporte.setText("Generar Reporte");
        btnGenerarReporte.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGenerarReporteActionPerformed(evt);
            }
        });

        lblIdCampania.setText("Id campaña:");

        txtIdCampania.setColumns(20);
        txtIdCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(5,1));
        txtIdCampania.addPropertyChangeListener(new
java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
```

```
                txtIdCampaniaPropertyChange(evt);
            }
        });
        txtIdCampania.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyReleased(java.awt.event.KeyEvent evt) {
                txtIdCampaniaKeyReleased(evt);
            }
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtIdCampaniaKeyTyped(evt);
            }
        });

        btnLOVCampania.setText("...");
        btnLOVCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnLOVCampaniaActionPerformed(evt);
            }
        });

        txtCampania.setEditable(false);

        lblCampania.setText("Campaña:");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(51, 51, 51)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblIdCampania)
                    .addComponent(lblCampania))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(btnGenerarReporte)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, 138,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(btnLOVCampania))
                    .addComponent(txtIdCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, 50,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(56, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(22, 22, 22)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblIdCampania)
```

```
                    .addComponent(txtIdCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnLOVCampania)
                    .addComponent(lblCampania))
                .addGap(33, 33, 33)
                .addComponent(btnGenerarReporte)
                .addContainerGap(23, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void btnGenerarReporteActionPerformed(java.awt.event.ActionEvent evt) {

        if (this.txtIdCampania.getText().trim().equals("")) {
            logger.info("El id de la campaña no puede ser nulo");
            JOptionPane.showMessageDialog(this, "El id de la campaña no puede ser
nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }

        Campania c = null;
        try {
            c =
CampaniaBO.buscarPorPK(Integer.parseInt(this.txtIdCampania.getText()));
        } catch (ErrorAplicativo ea) {
            ea.printStackTrace();
        }
        if (c == null) {
            JOptionPane.showMessageDialog(this, "La campaña ingresada no existe.",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }

        Map parametros = new HashMap();
        parametros.put("P_ID_CAMPANIA",
Integer.parseInt(this.txtIdCampania.getText()));
        try {
            logger.info("Inicio de ejecución Reporte de Bitácora de Camapaña");
            this.gestorReportes.mostrarReporte("ReporteBitacoraCampania", true,
parametros);
            logger.info("Fin de ejecución Reporte de Bitácora de Camapaña");
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }

    private void btnLOVCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        new LOVCampanias(this, true).setVisible(true);
    }

    private void txtIdCampaniaKeyReleased(java.awt.event.KeyEvent evt) {
        if (this.txtIdCampania.getText().equals("")) {
            return;
        }
        Campania c = null;
        try {
            c =
CampaniaBO.buscarPorPK(Integer.parseInt(this.txtIdCampania.getText()));
        } catch (ErrorAplicativo ea) {
            ea.printStackTrace();
        }
        if (c != null) {
            this.txtIdCampania.setText(String.valueOf(c.getIdCampania()));
            this.txtCampania.setText(c.getCampania());
        } else {
            this.txtCampania.setText("");
        }
    }

    private void txtIdCampaniaKeyTyped(java.awt.event.KeyEvent evt) {
    }

    private void txtIdCampaniaPropertyChange(java.beans.PropertyChangeEvent evt) {
    }

    public void setCampania(ConsultaCampanias campania) {
        this.campania = campania;
        this.txtIdCampania.setText(String.valueOf(this.campania.getIdCampania()));
        this.txtCampania.setText(this.campania.getCampania());
    }

    private void cargaDatos() {
        //Carga Gestor de Reportes
        this.gestorReportes = new GestorReportesBO();
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnGenerarReporte;
    private javax.swing.JButton btnLOVCampania;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblIdCampania;
    private javax.swing.JTextField txtCampania;
    private javax.swing.JTextField txtIdCampania;
    // End of variables declaration
    private ConsultaCampanias campania;
    private GestorReportesBO gestorReportes;
    private static Logger logger = Logger.getLogger(ReporteBitacoraCampania.class);
}
```

4.1.9.4 ReporteCostosCampanias

```
package com.jabluecatch.gui.reportes;
```

```
import com.jabluecatch.bo.GestorReportesBO;
import com.mydatepicker.main.DatePicker;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import java.sql.Timestamp;
import java.util.Calendar;
import java.util.Date;
import java.util.HashMap;
import java.util.Locale;
import java.util.Map;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de ejecución del reporte de costos de
 * campañas
 * @author Fas
 */
public class ReporteCostosCampanias extends javax.swing.JFrame {

    /** Creates new form ReporteUsuarios */
    public ReporteCostosCampanias() {
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnGenerarReporte = new javax.swing.JButton();
        lblFechaInicio = new javax.swing.JLabel();
        dtfFechaInicio = new com.jabluecatch.util.DateTextField();
        lblFechaFin = new javax.swing.JLabel();
        dtfFechaFin = new com.jabluecatch.util.DateTextField();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Reporte de Costos de Campaña");
        setBounds(new java.awt.Rectangle(350, 100, 370, 150));

        btnGenerarReporte.setText("Generar Reporte");
        btnGenerarReporte.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGenerarReporteActionPerformed(evt);
            }
        });

        lblFechaInicio.setText("Fecha inicio (dd/mm/yyyy):");

        dtfFechaInicio.setColumns(10);
```

```
        dtfFechaInicio.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaInicioMouseReleased(evt);
            }
        });

        lblFechaFin.setText("Fecha fin (dd/mm/yyyy):");

        dtfFechaFin.setColumns(10);
        dtfFechaFin.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                dtfFechaFinMouseReleased(evt);
            }
        });
        dtfFechaFin.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusLost(java.awt.event.FocusEvent evt) {
                dtfFechaFinFocusLost(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(121, 121, 121)
                        .addComponent(btnGenerarReporte))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(64, 64, 64)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(lblFechaFin)
                            .addComponent(lblFechaInicio))
                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(dtfFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(dtfFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(72, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(29, 29, 29)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaInicio)
                    .addComponent(dtfFechaInicio,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                .addGap(9, 9, 9)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblFechaFin)
                    .addComponent(dtfFechaFin,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(26, 26, 26)
                .addComponent(btnGenerarReporte)
                .addContainerGap(23, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void btnGenerarReporteActionPerformed(java.awt.event.ActionEvent evt) {

        if (this.dtfFechaInicio.getText().trim().equals("")) {
            logger.info("La fecha de inicio no puede ser nula");
            JOptionPane.showMessageDialog(this, "La fecha de inicio no puede ser
nula", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        } else if (this.dtfFechaFin.getText().trim().equals("")) {

this.dtfFechaFin.setText(FormateadorFecha.aJavaDateString(Calendar.getInstance().ge
tTime()));
        }

        //Convertir las fechas
        String sFechaInicio = this.dtfFechaInicio.getText() + " 00:00:00";
        String sFechaFin = this.dtfFechaFin.getText() + " 23:59:59";

        Date dFechaInicio = null;
        Date dFechaFin = null;
        try {
            dFechaInicio = FormateadorFecha.aJavaDateTimeDate(sFechaInicio);
            dFechaFin = FormateadorFecha.aJavaDateTimeDate(sFechaFin);
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Información",
JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        if (dFechaFin.before(dFechaInicio)) {
            logger.info("La fecha de finalización no puede ser menor a la fecha de
inicio");
            JOptionPane.showMessageDialog(this, "La fecha de finalización no puede
ser menor a la fecha de inicio", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
        if (dFechaFin.equals(dFechaInicio)) {
            logger.info("La fecha de finalización no puede ser igual a la fecha de
inicio");
            JOptionPane.showMessageDialog(this, "La fecha de finalización no puede
ser igual a la fecha de inicio", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }
```

```
        Timestamp fechaInicio = new Timestamp(dFechaInicio.getTime());
        Timestamp fechaFin = new Timestamp(dFechaFin.getTime());

        Map parametros = new HashMap();
        parametros.put("P_FECHA_INICIO", fechaInicio);
        parametros.put("P_FECHA_FIN", fechaFin);
        try {
            logger.info("Inicio de ejecución Reporte de Costos de Camapañas");
            this.gestorReportes.mostrarReporte("ReporteCostosCampanias", true,
parametros);
            logger.info("Fin de ejecución Reporte de Costos de Camapañas");
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }

    private void dtfFechaFinFocusLost(java.awt.event.FocusEvent evt) {
}

    private void dtfFechaInicioMouseReleased(java.awt.event.MouseEvent evt) {
        DatePicker dp = new DatePicker(dtfFechaInicio, new Locale("ES"));
        Date fechaSeleccionada = dp.parseDate(dtfFechaInicio.getText());
        dp.setSelectedDate(fechaSeleccionada);
        dp.start(dtfFechaInicio);
    }

    private void dtfFechaFinMouseReleased(java.awt.event.MouseEvent evt) {
        DatePicker dp = new DatePicker(dtfFechaFin, new Locale("ES"));
        Date fechaSeleccionada = dp.parseDate(dtfFechaFin.getText());
        dp.setSelectedDate(fechaSeleccionada);
        dp.start(dtfFechaFin);
    }

    private void cargaDatos() {
        //Carga Gestor de Reportes
        this.gestorReportes = new GestorReportesBO();
        this.dtfFechaInicio.setText("01/01/2010");

this.dtfFechaFin.setText(FormateadorFecha.aJavaDateString(Calendar.getInstance().ge
tTime()));
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnGenerarReporte;
    private com.jabluecatch.util.DateTextField dtfFechaFin;
    private com.jabluecatch.util.DateTextField dtfFechaInicio;
    private javax.swing.JLabel lblFechaFin;
    private javax.swing.JLabel lblFechaInicio;
    // End of variables declaration
    private GestorReportesBO gestorReportes;
    private static Logger logger = Logger.getLogger(ReporteCostosCampanias.class);
}
```

4.1.9.5 ReporteImpactoCampania

```
package com.jabluecatch.gui.reportes;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.bo.GestorReportesBO;
import com.jabluecatch.dto.ConsultaCampanias;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.util.ErrorAplicativo;
import java.util.HashMap;
import java.util.Map;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de ejecución del reporte de impacto de
 * campaña
 * @author Fas
 */
public class ReporteImpactoCampania extends javax.swing.JFrame implements
CampaniaParametrizable {

    /** Creates new form ReporteUsuarios */
    public ReporteImpactoCampania() {
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnGenerarReporte = new javax.swing.JButton();
        lblIdCampania = new javax.swing.JLabel();
        txtIdCampania = new javax.swing.JTextField();
        btnLOVCampania = new javax.swing.JButton();
        txtCampania = new javax.swing.JTextField();
        lblCampania = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Reporte de Impacto de Campaña");
        setBounds(new java.awt.Rectangle(350, 100, 370, 150));

        btnGenerarReporte.setText("Generar Reporte");
        btnGenerarReporte.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGenerarReporteActionPerformed(evt);
            }
        });

        lblIdCampania.setText("Id campaña:");
```

```
        txtIdCampania.setColumns(20);
        txtIdCampania.setDocument(new
com.jabluecatch.util.FormateadorDocumento(5,1));
        txtIdCampania.addPropertyChangeListener(new
java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                txtIdCampaniaPropertyChange(evt);
            }
        });
        txtIdCampania.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyReleased(java.awt.event.KeyEvent evt) {
                txtIdCampaniaKeyReleased(evt);
            }
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtIdCampaniaKeyTyped(evt);
            }
        });

        btnLOVCampania.setText("...");
        btnLOVCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnLOVCampaniaActionPerformed(evt);
            }
        });

        txtCampania.setEditable(false);

        lblCampania.setText("Campaña:");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(51, 51, 51)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblIdCampania)
                    .addComponent(lblCampania))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(btnGenerarReporte)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, 138,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(btnLOVCampania))
                    .addComponent(txtIdCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, 50,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(56, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
```

```
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(22, 22, 22)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblIdCampania)
                    .addComponent(txtIdCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtCampania,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnLOVCampania)
                    .addComponent(lblCampania))
                .addGap(33, 33, 33)
                .addComponent(btnGenerarReporte)
                .addContainerGap(23, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void btnGenerarReporteActionPerformed(java.awt.event.ActionEvent evt) {

        if (this.txtIdCampania.getText().trim().equals("")) {
            logger.info("El id de la campaña no puede ser nulo");
            JOptionPane.showMessageDialog(this, "El id de la campaña no puede ser
nulo", "Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }

        Campania c = null;
        try {
            c =
CampaniaBO.buscarPorPK(Integer.parseInt(this.txtIdCampania.getText()));
        } catch (ErrorAplicativo ea) {
            ea.printStackTrace();
        }
        if (c == null) {
            JOptionPane.showMessageDialog(this, "La campaña ingresada no existe.",
"Información", JOptionPane.INFORMATION_MESSAGE);
            return;
        }

        Map parametros = new HashMap();
        parametros.put("P_ID_CAMPANIA",
Integer.parseInt(this.txtIdCampania.getText()));
        try {
            logger.info("Inicio de ejecución Reporte de Impacto de Camapaña");
            this.gestorReportes.mostrarReporte("ReporteImpactoCampania", true,
parametros);
            logger.info("Fin de ejecución Reporte de Impacto de Camapaña");
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
```

```
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }

    private void btnLOVCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        new LOVCampanias(this, true).setVisible(true);
    }

    private void txtIdCampaniaKeyReleased(java.awt.event.KeyEvent evt) {
        if (this.txtIdCampania.getText().equals("")) {
            return;
        }
        Campania c = null;
        try {
            c =
CampaniaBO.buscarPorPK(Integer.parseInt(this.txtIdCampania.getText()));
        } catch (ErrorAplicativo ea) {
            ea.printStackTrace();
        }
        if (c != null) {
            this.txtIdCampania.setText(String.valueOf(c.getIdCampania()));
            this.txtCampania.setText(c.getCampania());
        } else {
            this.txtCampania.setText("");
        }
    }

    private void txtIdCampaniaKeyTyped(java.awt.event.KeyEvent evt) {
    }

    private void txtIdCampaniaPropertyChange(java.beans.PropertyChangeEvent evt) {
    }

    public void setCampania(ConsultaCampanias campania) {
        this.campania = campania;
        this.txtIdCampania.setText(String.valueOf(this.campania.getIdCampania()));
        this.txtCampania.setText(this.campania.getCampania());
    }

    private void cargaDatos() {
        //Carga Gestor de Reportes
        this.gestorReportes = new GestorReportesBO();
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnGenerarReporte;
    private javax.swing.JButton btnLOVCampania;
    private javax.swing.JLabel lblCampania;
    private javax.swing.JLabel lblIdCampania;
    private javax.swing.JTextField txtCampania;
    private javax.swing.JTextField txtIdCampania;
    // End of variables declaration
    private ConsultaCampanias campania;
    private GestorReportesBO gestorReportes;
    private static Logger logger = Logger.getLogger(ReporteImpactoCampania.class);
}
```

4.1.9.6 ReporteUsuarios

```
package com.jabluecatch.gui.reportes;

import com.jabluecatch.bo.GestorReportesBO;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import java.util.Calendar;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de ejecución del reporte de usuarios del
 * sistema
 * @author Fausto Almeida
 */
public class ReporteUsuarios extends javax.swing.JDialog {

    /** Creates new form ReporteUsuarios */
    public ReporteUsuarios() {
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnGenerarReporte = new javax.swing.JButton();
        lblReporteHasta = new javax.swing.JLabel();
        lblHasta = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Reporte de Usuarios");
        setBounds(new java.awt.Rectangle(350, 100, 355, 155));

        btnGenerarReporte.setText("Generar Reporte");
        btnGenerarReporte.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGenerarReporteActionPerformed(evt);
            }
        });

        lblReporteHasta.setText("Reporte de usuarios hasta:");

        lblHasta.setText("l");
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(29, 29, 29)
                .addComponent(lblReporteHasta)
                .addGap(18, 18, 18)
                .addComponent(lblHasta)
                .addContainerGap(175, Short.MAX_VALUE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(123, Short.MAX_VALUE)
                .addComponent(btnGenerarReporte)
                .addGap(119, 119, 119))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(22, 22, 22)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(lblReporteHasta)
                    .addComponent(lblHasta))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 46,
Short.MAX_VALUE)
                .addComponent(btnGenerarReporte)
                .addGap(27, 27, 27))
        );

        pack();
    }// </editor-fold>

    private void btnGenerarReporteActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            logger.info("Inicio de ejecución Reporte de Usuarios");
            this.gestorReportes.mostrarReporte("ReporteUsuarios", true, null);
            logger.info("Fin de ejecución Reporte de Usuarios");
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }

    private void cargaDatos() {
        //Carga Gestor de Reportes
        this.gestorReportes = new GestorReportesBO();

this.lblHasta.setText(FormateadorFecha.aJavaDateString(Calendar.getInstance().getTi
me()));

    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnGenerarReporte;
    private javax.swing.JLabel lblHasta;
```

```
    private javax.swing.JLabel lblReporteHasta;
    // End of variables declaration
    private GestorReportesBO gestorReportes;
    private static Logger logger = Logger.getLogger(ReporteUsuarios.class);
}
```

4.1.10 Paquete com.jabluecatch.gui

4.1.10.1 AcercaDe

```
package com.jabluecatch.gui;

import javax.swing.JFrame;

/**
 * Clase que representa la ventana de Acerca de
 * @author Fausto Almeida
 */
public class AcercaDe extends javax.swing.JDialog {

    /** Creates new form AcercaDe */
    public AcercaDe(JFrame padre, boolean modal) {
        super(padre, modal);
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        jLabel10 = new javax.swing.JLabel();
        jLabel11 = new javax.swing.JLabel();
        btnAceptar = new javax.swing.JButton();
```

```
        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("BlueCatch v0.1 - Acerca de");
        setBounds(new java.awt.Rectangle(390, 390, 400, 320));
        setResizable(false);

        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18));
        jLabel1.setText("BlueCatch v0.1");

        jLabel2.setFont(new java.awt.Font("Tahoma", 0, 12));
        jLabel2.setText("Prototipo de Sistema de Envío de Publicidad Vía
Bluetooth");

        jLabel3.setFont(new java.awt.Font("Tahoma", 0, 12));
        jLabel3.setText("Proyecto de Curso de Graduación");

        jLabel4.setText("Institución:");

        jLabel5.setText("Desarrollado por:");

        jLabel6.setText("Profesor Guía:");

        jLabel7.setText("Fecha:");

        jLabel8.setText("Fausto S. Almeida Campos");

        jLabel9.setText("Universidad de Guayaquil - CISC");

        jLabel10.setText("Ing. David Benavides");

        jLabel11.setText("Marzo 2011");

        btnAceptar.setText("Aceptar");
        btnAceptar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAceptarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel3)
                            .addComponent(jLabel2)
                            .addComponent(jLabel1)
                            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(jLabel5)
```

```
                                    .addComponent(jLabel4)
                                    .addComponent(jLabel6)
                                    .addComponent(jLabel7))
                                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                    .addComponent(jLabel11)
                                    .addComponent(jLabel10)
                                    .addComponent(jLabel9)
                                    .addComponent(jLabel8)))))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(152, 152, 152)
                        .addComponent(btnAceptar)))
                .addContainerGap(44, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(27, 27, 27)
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)
                .addGap(18, 18, 18)
                .addComponent(jLabel3)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel4)
                    .addComponent(jLabel9))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel5)
                    .addComponent(jLabel8))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel6)
                    .addComponent(jLabel10))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel7)
                    .addComponent(jLabel11))
                .addGap(26, 26, 26)
                .addComponent(btnAceptar)
                .addContainerGap(28, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAceptar;
    private javax.swing.JLabel jLabel1;
```

```
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    // End of variables declaration
}
```

4.1.10.2 Login

```
package com.jabluecatch.gui;

import com.jabluecatch.bo.UsuarioBO;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de login
 * @author Fas
 */
public class Login extends javax.swing.JFrame {

    /** Creates new form Login */
    public Login() {
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        lblUsuario = new javax.swing.JLabel();
        lblClave = new javax.swing.JLabel();
        txtUsuario = new javax.swing.JTextField();
        btnAceptar = new javax.swing.JButton();
        btnCancelar = new javax.swing.JButton();
        txtClave = new javax.swing.JPasswordField();
        lblLogo = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
```

```
        setTitle("BlueCatch v0.1 - Login");
        setBounds(new java.awt.Rectangle(300, 300, 320, 150));
        setResizable(false);

        lblUsuario.setText("Usuario:");

        lblClave.setText("Clave:");

        txtUsuario.setColumns(12);
        txtUsuario.setDocument(new
com.jabluecatch.util.FormateadorDocumento(15,2));
        txtUsuario.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtUsuarioActionPerformed(evt);
            }
        });

        btnAceptar.setText("Aceptar");
        btnAceptar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAceptarActionPerformed(evt);
            }
        });

        btnCancelar.setText("Cancelar");
        btnCancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnCancelarActionPerformed(evt);
            }
        });

        txtClave.setColumns(12);
        txtClave.setDocument(new com.jabluecatch.util.FormateadorDocumento(15,0));
        txtClave.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtClaveActionPerformed(evt);
            }
        });

        lblLogo.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
gin.png"))); // NOI18N

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(28, 28, 28)
                .addComponent(lblLogo)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(lblClave)
                    .addComponent(lblUsuario))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(txtClave, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(37, Short.MAX_VALUE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(79, Short.MAX_VALUE)
                .addComponent(btnAceptar)
                .addGap(18, 18, 18)
                .addComponent(btnCancelar)
                .addGap(64, 64, 64))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(33, 33, 33)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(lblUsuario)
                            .addComponent(txtUsuario,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(8, 8, 8)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(lblClave)
                            .addComponent(txtClave,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(25, 25, 25)
                        .addComponent(lblLogo)))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnCancelar)
                    .addComponent(btnAceptar))
                .addContainerGap(25, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void txtUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
        btnAceptar.doClick();
    }

    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }
```

```
    private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
        try {

            final Usuario result =
UsuarioBO.validarInicioSesion(this.txtUsuario.getText(), new
String(this.txtClave.getPassword()));
            if (result != null) {
                //abrir ventana principal
                java.awt.EventQueue.invokeLater(new Runnable() {

                    public void run() {
                        new PantallaPrincipal(result).setVisible(true);
                    }
                });
                this.dispose();
            }
        } catch (ErrorAplicativo ea) {
            if (ea.getTecnico() != null) {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            } else {
                logger.warn(ea.getErrorTecnico());
                JOptionPane.showMessageDialog(this, "El usuario o la contraseña
ingresada no son válidas.", "Error", JOptionPane.ERROR_MESSAGE);
            }
            this.txtClave.setText("");
        }
    }

    private void txtClaveActionPerformed(java.awt.event.ActionEvent evt) {
        this.btnAceptar.doClick();
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAceptar;
    private javax.swing.JButton btnCancelar;
    private javax.swing.JLabel lblClave;
    private javax.swing.JLabel lblLogo;
    private javax.swing.JLabel lblUsuario;
    private javax.swing.JPasswordField txtClave;
    private javax.swing.JTextField txtUsuario;
    // End of variables declaration
    private static Logger logger = Logger.getLogger(Login.class);
}
```

4.1.10.3 PantallaPrincipal

```
package com.jabluecatch.gui;

import com.jabluecatch.bo.CampaniaBO;
import com.jabluecatch.bo.DetalleCampaniaBO;
import com.jabluecatch.comm.DetectorDispositivosRemotos;
import com.jabluecatch.comm.Despachador;
import com.jabluecatch.gui.campania.ConsultarCampanias;
import com.jabluecatch.gui.campania.CrearCampania;
```

```
import com.jabluecatch.gui.campania.VerDatosCampania;
import com.jabluecatch.gui.sistema.CambiarClave;
import com.jabluecatch.gui.sistema.ConsultarUsuarios;
import com.jabluecatch.gui.sistema.ParametrosGenerales;
import com.jabluecatch.gui.sistema.CrearUsuario;
import com.jabluecatch.entidades.Campania;
import com.jabluecatch.dto.Tarea;
import com.jabluecatch.entidades.Usuario;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.FormateadorFecha;
import com.jabluecatch.gui.reportes.ReporteBitacoraCampania;
import com.jabluecatch.gui.reportes.ReporteUsuarios;
import com.jabluecatch.entidades.DetalleCampania;
import com.jabluecatch.gui.reportes.ReporteCostosCampanias;
import com.jabluecatch.gui.reportes.ReporteImpactoCampania;
import com.jabluecatch.util.ModeloTablaMonitor;
import java.io.File;
import java.util.Vector;
import javax.swing.JOptionPane;
import org.apache.log4j.Logger;

/**
 * Clase que representa la ventana de la pantalla principal, el monitor de
 * tareas
 * @author Fas
 */
public class PantallaPrincipal extends javax.swing.JFrame {

    /** Creates new form PantallaPrincipal */
    public PantallaPrincipal(Usuario usuarioSesion) {
        initComponents();
        this.setIconImage(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/lo
go.png")).getImage());
        this.usuarioSesion = usuarioSesion;
        cargaDatos();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        txtCampaniaCA = new javax.swing.JTextField();
        lblCampaniaCA = new javax.swing.JLabel();
        lblDesdeCA = new javax.swing.JLabel();
        txtDesdeCA = new javax.swing.JTextField();
        lblHastaCA = new javax.swing.JLabel();
        txtHastaCA = new javax.swing.JTextField();
        btnMasCA = new javax.swing.JButton();
        lblCampaniaSC = new javax.swing.JLabel();
        txtCampaniaSC = new javax.swing.JTextField();
        lblDesdeSC = new javax.swing.JLabel();
        txtDesdeSC = new javax.swing.JTextField();
```

```
lblHastaSC = new javax.swing.JLabel();
txtHastaSC = new javax.swing.JTextField();
btnMasSC = new javax.swing.JButton();
btnIniciar = new javax.swing.JButton();
btnDetener = new javax.swing.JButton();
jScrollPane1 = new javax.swing.JScrollPane();
tabMonitor = new javax.swing.JTable();
btnRecargar = new javax.swing.JButton();
chkReenviarExitosos = new javax.swing.JCheckBox();
jMenuBar1 = new javax.swing.JMenuBar();
mnuSistema = new javax.swing.JMenu();
mitCrearUsuario = new javax.swing.JMenuItem();
mitConsultarUsuarios = new javax.swing.JMenuItem();
mitCambiarClave = new javax.swing.JMenuItem();
jSeparator1 = new javax.swing.JPopupMenu.Separator();
mitParametrosGen = new javax.swing.JMenuItem();
mitSalir = new javax.swing.JMenuItem();
mnuCampania = new javax.swing.JMenu();
mitCrearCampania = new javax.swing.JMenuItem();
mitConsultarCampanias = new javax.swing.JMenuItem();
mnuReportes = new javax.swing.JMenu();
mitReporteUsuarios = new javax.swing.JMenuItem();
mitReporteBitacoraCampania = new javax.swing.JMenuItem();
mitReporteImpactoCampania = new javax.swing.JMenuItem();
mitReporteCostosCampanias = new javax.swing.JMenuItem();
mnuAyuda = new javax.swing.JMenu();
mitAcercaDe = new javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setTitle("BlueCatch v0.1");
setBounds(new java.awt.Rectangle(150, 150, 800, 600));
setResizable(false);

txtCampaniaCA.setFocusable(false);

lblCampaniaCA.setText("Campaña activa:");

lblDesdeCA.setText("Desde:");

txtDesdeCA.setColumns(13);
txtDesdeCA.setFocusable(false);

lblHastaCA.setText("Hasta:");

txtHastaCA.setColumns(13);
txtHastaCA.setFocusable(false);

btnMasCA.setText("+");
btnMasCA.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnMasCAActionPerformed(evt);
    }
});

lblCampaniaSC.setText("Siguiente campaña:");

txtCampaniaSC.setFocusable(false);
```

```
        lblDesdeSC.setText("Desde:");

        txtDesdeSC.setColumns(13);
        txtDesdeSC.setFocusable(false);

        lblHastaSC.setText("Hasta:");

        txtHastaSC.setColumns(13);
        txtHastaSC.setFocusable(false);

        btnMasSC.setText("+");
        btnMasSC.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnMasSCActionPerformed(evt);
            }
        });

        btnIniciar.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/ru
n.png"))); // NOI18N
        btnIniciar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnIniciarActionPerformed(evt);
            }
        });

        btnDetener.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/st
op.png"))); // NOI18N
        btnDetener.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnDetenerActionPerformed(evt);
            }
        });

        tabMonitor.setModel(new ModeloTablaMonitor());
        jScrollPane1.setViewportView(tabMonitor);

        btnRecargar.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/com/jabluecatch/recursos/imagenes/re
load.png"))); // NOI18N
        btnRecargar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnRecargarActionPerformed(evt);
            }
        });

        chkReenviarExitosos.setText("Reenvio constante de contenido a
dispositivos.");
        chkReenviarExitosos.setToolTipText("Reenviar contenido multimedia a los
dispositivos que ya lo han recibido satisfactoriamente.");

        mnuSistema.setText("Sistema");
        mnuSistema.setMaximumSize(new java.awt.Dimension(58, 32767));
        mnuSistema.setPreferredSize(new java.awt.Dimension(59, 19));

        mitCrearUsuario.setText("Crear usuario");
        mitCrearUsuario.addActionListener(new java.awt.event.ActionListener() {
```

```
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitCrearUsuarioActionPerformed(evt);
            }
        });
        mnuSistema.add(mitCrearUsuario);

        mitConsultarUsuarios.setText("Consultar usuarios");
        mitConsultarUsuarios.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitConsultarUsuariosActionPerformed(evt);
            }
        });
        mnuSistema.add(mitConsultarUsuarios);

        mitCambiarClave.setText("Cambiar clave");
        mitCambiarClave.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitCambiarClaveActionPerformed(evt);
            }
        });
        mnuSistema.add(mitCambiarClave);
        mnuSistema.add(jSeparator1);

        mitParametrosGen.setText("Parámetros generales");
        mitParametrosGen.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitParametrosGenActionPerformed(evt);
            }
        });
        mnuSistema.add(mitParametrosGen);

        mitSalir.setText("Salir");
        mitSalir.setPreferredSize(new java.awt.Dimension(59, 19));
        mitSalir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitSalirActionPerformed(evt);
            }
        });
        mnuSistema.add(mitSalir);

        jMenuBar1.add(mnuSistema);

        mnuCampania.setText("Campaña");
        mnuCampania.setMaximumSize(new java.awt.Dimension(65, 32767));
        mnuCampania.setPreferredSize(new java.awt.Dimension(69, 19));


mitCrearCampania.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.K
eyEvent.VK_N, java.awt.event.InputEvent.CTRL_MASK));
        mitCrearCampania.setText("Crear campaña");
        mitCrearCampania.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitCrearCampaniaActionPerformed(evt);
            }
        });
        mnuCampania.add(mitCrearCampania);
```

```
        mitConsultarCampanias.setText("Consultar campañas");
        mitConsultarCampanias.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitConsultarCampaniasActionPerformed(evt);
            }
        });
        mnuCampania.add(mitConsultarCampanias);

        jMenuBar1.add(mnuCampania);

        mnuReportes.setText("Reportes");
        mnuReportes.setMaximumSize(new java.awt.Dimension(65, 32767));

        mitReporteUsuarios.setText("Reporte de usuarios");
        mitReporteUsuarios.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitReporteUsuariosActionPerformed(evt);
            }
        });
        mnuReportes.add(mitReporteUsuarios);

        mitReporteBitacoraCampania.setText("Reporte de Bitácora por Campaña");
        mitReporteBitacoraCampania.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitReporteBitacoraCampaniaActionPerformed(evt);
            }
        });
        mnuReportes.add(mitReporteBitacoraCampania);

        mitReporteImpactoCampania.setText("Reporte de Impacto de Campaña");
        mitReporteImpactoCampania.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitReporteImpactoCampaniaActionPerformed(evt);
            }
        });
        mnuReportes.add(mitReporteImpactoCampania);

        mitReporteCostosCampanias.setText("Reporte de Costos de Campañas");
        mitReporteCostosCampanias.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitReporteCostosCampaniasActionPerformed(evt);
            }
        });
        mnuReportes.add(mitReporteCostosCampanias);

        jMenuBar1.add(mnuReportes);

        mnuAyuda.setText("Ayuda");
        mnuAyuda.setMaximumSize(new java.awt.Dimension(58, 32767));
        mnuAyuda.setPreferredSize(new java.awt.Dimension(59, 19));


mitAcercaDe.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEve
nt.VK_F1, 0));
```

```
        mitAcercaDe.setText("Acerca de...");
        mitAcercaDe.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                mitAcercaDeActionPerformed(evt);
            }
        });
        mnuAyuda.add(mitAcercaDe);

        jMenuBar1.add(mnuAyuda);

        setJMenuBar(jMenuBar1);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(51, 51, 51)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(lblCampaniaCA)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
false)
                                .addGroup(layout.createSequentialGroup()
                                    .addComponent(txtCampaniaCA)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(btnMasCA))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                        .addComponent(btnDetener)
                                        .addGroup(layout.createSequentialGroup()
                                            .addComponent(lblDesdeCA)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                                .addComponent(txtDesdeCA,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                                    .addGap(24, 24, 24)
                                    .addComponent(lblHastaCA)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(txtHastaCA,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 58,
Short.MAX_VALUE)
                        .addComponent(btnRecargar)
                        .addGap(51, 51, 51)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(lblCampaniaSC)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
false)
                                .addGroup(layout.createSequentialGroup()
                                    .addComponent(txtCampaniaSC)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(btnMasSC))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()
                                    .addComponent(lblDesdeSC)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(txtDesdeSC,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addGap(18, 18, 18)
                                    .addComponent(lblHastaSC)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(txtHastaSC,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addComponent(chkReenviarExitosos,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
                        .addGap(55, 55, 55))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(btnIniciar)
                        .addContainerGap(794, Short.MAX_VALUE))))
            .addGroup(layout.createSequentialGroup()
                .addGap(32, 32, 32)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
835, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(35, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(24, 24, 24)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(lblCampaniaCA)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(txtCampaniaCA,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(btnMasCA))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(lblDesdeCA)
                            .addComponent(lblHastaCA)
                            .addComponent(txtDesdeCA,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtHastaCA,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(lblCampaniaSC)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(btnRecargar)
                            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                                    .addComponent(txtCampaniaSC,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(btnMasSC))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                                    .addComponent(lblDesdeSC)
                                    .addComponent(txtDesdeSC,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(lblHastaSC)
                                    .addComponent(txtHastaSC,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(btnIniciar)
                    .addComponent(btnDetener)
                    .addComponent(chkReenviarExitosos))
                .addGap(18, 18, 18)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(28, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void mitSalirActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        System.exit(0);
    }

    private void mitConsultarCampaniasActionPerformed(java.awt.event.ActionEvent
evt) {
```

```
        //abrir ventana
        new ConsultarCampanias(this, true).setVisible(true);

    }

    private void mitParametrosGenActionPerformed(java.awt.event.ActionEvent evt) {
        new ParametrosGenerales(this, true).setVisible(true);
    }

    private void mitCrearUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
        //abrir ventana
        new CrearUsuario(this, true).setVisible(true);
    }

    private void mitCambiarClaveActionPerformed(java.awt.event.ActionEvent evt) {
        //abrir ventana
        new CambiarClave(this, true,
this.usuarioSesion.getUsuario()).setVisible(true);
    }

    private void mitConsultarUsuariosActionPerformed(java.awt.event.ActionEvent
evt) {
        //abrir ventana
        new ConsultarUsuarios(this, true).setVisible(true);
    }

    private void mitCrearCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        //abrir ventana
        new CrearCampania(this, true).setVisible(true);
    }

    private void mitAcercaDeActionPerformed(java.awt.event.ActionEvent evt) {
        //abrir ventana
        new AcercaDe(this, true).setVisible(true);
    }

    private void btnMasCAActionPerformed(java.awt.event.ActionEvent evt) {
        new VerDatosCampania(this, false,
this.campaniaCA.getIdCampania()).setVisible(true);
    }

    private void btnMasSCActionPerformed(java.awt.event.ActionEvent evt) {
        new VerDatosCampania(this, false,
this.campaniaSC.getIdCampania()).setVisible(true);
    }

    private void btnRecargarActionPerformed(java.awt.event.ActionEvent evt) {
        this.btnRecargar.setEnabled(false);
        this.cargarCampanias();
        this.procesoEnvio.cambiarCampania(this.campaniaCA, this.detallesCampania);

        this.btnRecargar.setEnabled(true);
    }

    private void btnIniciarActionPerformed(java.awt.event.ActionEvent evt) {

        this.btnRecargar.doClick();
```

```
        if (this.detallesCampania == null) {
            logger.warn("btnIniciarActionPerformed - No se puede iniciar el envío
de contenido porque no existe una campaña activa.");
            JOptionPane.showMessageDialog(this, "No se puede iniciar el envío de
contenido porque no existe una campaña activa", "Error",
JOptionPane.ERROR_MESSAGE);
            return;
        }


        //Verificar que los archivos a enviar existan
        for (DetalleCampania dc : this.detallesCampania) {
            File f = new File(dc.getRuta());
            if (!f.exists()) {
                logger.warn("btnIniciarActionPerformed - El archivo <" +
dc.getRuta().replace("\\\\", "\\") + "> no existe.");
                JOptionPane.showMessageDialog(this, "El archivo <" +
dc.getRuta().replace("\\\\", "\\") + "> no existe", "Error",
JOptionPane.ERROR_MESSAGE);
                return;
            }
        }

        this.btnIniciar.setEnabled(false);
        try {
            new DetectorDispositivosRemotos().iniciar();
        } catch (ErrorAplicativo ea) {
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
            this.btnIniciar.setEnabled(true);
            return;
        }
        logger.info("btnIniciarActionPerformed - Reanudando proceso de envío de
publicidad.");
        this.procesoEnvio.setContinuar(true);
        this.btnDetener.setEnabled(true);


    }

    private void btnDetenerActionPerformed(java.awt.event.ActionEvent evt) {

        logger.info("btnIniciarActionPerformed - Deteniendo proceso de envío de
publicidad.");
        this.btnDetener.setEnabled(false);
        this.procesoEnvio.setContinuar(false);
        this.btnIniciar.setEnabled(true);

    }

    private void mitReporteUsuariosActionPerformed(java.awt.event.ActionEvent evt)
{
        new ReporteUsuarios().setVisible(true);
    }

    private void
mitReporteBitacoraCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        new ReporteBitacoraCampania().setVisible(true);
    }

    private void
mitReporteImpactoCampaniaActionPerformed(java.awt.event.ActionEvent evt) {
        new ReporteImpactoCampania().setVisible(true);
    }

    private void
mitReporteCostosCampaniasActionPerformed(java.awt.event.ActionEvent evt) {
        new ReporteCostosCampanias().setVisible(true);
    }

    private void seteaAnchoColumnas() {
        this.tabMonitor.getColumn("Nombre dispositivo").setPreferredWidth(100);
        this.tabMonitor.getColumn("Archivo a enviar").setPreferredWidth(350);
        this.tabMonitor.getColumn("Estado").setPreferredWidth(150);
        this.tabMonitor.getColumn("Porcentaje completado").setPreferredWidth(100);
    }

    public synchronized void actualizarRegistroMonitor(Tarea rm) {
        for (Tarea t : this.getMonitor()) {
            if (rm.getIdTarea() == t.getIdTarea()) {
                t.setPorcentajeCompletado(rm.getPorcentajeCompletado());
                t.setEstado(rm.getEstado());
            }
        }
        this.tabMonitor.setModel(new ModeloTablaMonitor(this.getMonitor()));
        this.seteaAnchoColumnas();
    }

    public void agregarRegistroMonitor(Tarea rm) {
        for (Tarea t : this.getMonitor()) {
            if (rm.getIdTarea() == t.getIdTarea()) {
                return;
            }
        }
        this.getMonitor().add(rm);
        this.tabMonitor.setModel(new ModeloTablaMonitor(this.getMonitor()));
        this.seteaAnchoColumnas();
    }

    public void reiniciarBotones() {
        this.btnIniciar.setEnabled(true);
        this.btnDetener.setEnabled(false);
    }

    public boolean reenviarExitosos(){
        return this.chkReenviarExitosos.isSelected();
    }

    public void cargarCampanias() {
        //Cargar campania actual y campania siguiente
        try {
            CampaniaBO.conciliarFases();
            this.campaniaCA = CampaniaBO.obtenerCampaniaEnEjecucion();
            this.campaniaSC = CampaniaBO.obtenerSiguienteCampania();
        } catch (ErrorAplicativo ea) {
```

```
            logger.warn(ea.getErrorTecnico());
            JOptionPane.showMessageDialog(this, ea.getError(), "Error",
JOptionPane.ERROR_MESSAGE);
        }

        //Cargar sus datos
        this.txtCampaniaCA.setText((this.campaniaCA != null) ?
this.campaniaCA.getCampania() : "N/A");
        this.txtDesdeCA.setText((this.campaniaCA != null) ?
FormateadorFecha.aJavaDateTimeString(this.campaniaCA.getFechaInicio()) : "N/A");
        this.txtHastaCA.setText((this.campaniaCA != null) ?
(this.campaniaCA.getFechaFin() != null ?
FormateadorFecha.aJavaDateTimeString(this.campaniaCA.getFechaFin()) : "N/A") :
"N/A");
        this.btnMasCA.setEnabled(this.campaniaCA != null);
        this.btnIniciar.setEnabled(this.campaniaCA != null);
        this.btnDetener.setEnabled(false);

        this.txtCampaniaSC.setText((this.campaniaSC != null) ?
this.campaniaSC.getCampania() : "N/A");
        this.txtDesdeSC.setText((this.campaniaSC != null) ?
FormateadorFecha.aJavaDateTimeString(this.campaniaSC.getFechaInicio()) : "N/A");
        this.txtHastaSC.setText((this.campaniaSC != null) ?
(this.campaniaSC.getFechaFin() != null ?
FormateadorFecha.aJavaDateTimeString(this.campaniaSC.getFechaFin()) : "N/A") :
"N/A");
        this.btnMasSC.setEnabled(this.campaniaSC != null);


        if (this.campaniaCA != null) {
            try {
                this.detallesCampania =
DetalleCampaniaBO.buscarPorIdCampania(this.campaniaCA.getIdCampania());
            } catch (ErrorAplicativo ea) {
                logger.warn(ea.getErrorTecnico());
            }
        } else {
            this.detallesCampania = null;
        }

    }

    public void mostrarMensaje(String mensaje) {
        JOptionPane.showMessageDialog(this, mensaje, "Información",
JOptionPane.ERROR_MESSAGE);
    }

    private void cargaDatos() {
        //Inhabilitar opciones por rol
        if (this.usuarioSesion.getIdRol() != 1) {
            this.mitCrearUsuario.setEnabled(false);
            this.mitConsultarUsuarios.setEnabled(false);
            this.mitParametrosGen.setEnabled(false);
            this.mitCrearCampania.setEnabled(false);
            this.mitReporteUsuarios.setEnabled(false);
            this.mitReporteBitacoraCampania.setEnabled(false);
        }
        this.monitor = new Vector();
```

```
        this.seteaAnchoColumnas();

        //Carga los datos de las campanias actual y siguiente
        this.cargarCampanias();


        //Configurar proceso
        this.procesoEnvio = new Despachador(this, this.campaniaCA,
this.detallesCampania);
        this.procesoEnvio.start();

    }

    /**
     * @return the usuarioSesion
     */
    public Usuario getUsuarioSesion() {
        return usuarioSesion;
    }

    /**
     * @param usuarioSesion the usuarioSesion to set
     */
    public void setUsuarioSesion(Usuario usuarioSesion) {
        this.usuarioSesion = usuarioSesion;
    }

    /**
     * @return the monitor
     */
    public Vector<Tarea> getMonitor() {
        return monitor;
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnDetener;
    private javax.swing.JButton btnIniciar;
    private javax.swing.JButton btnMasCA;
    private javax.swing.JButton btnMasSC;
    private javax.swing.JButton btnRecargar;
    private javax.swing.JCheckBox chkReenviarExitosos;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JPopupMenu.Separator jSeparator1;
    private javax.swing.JLabel lblCampaniaCA;
    private javax.swing.JLabel lblCampaniaSC;
    private javax.swing.JLabel lblDesdeCA;
    private javax.swing.JLabel lblDesdeSC;
    private javax.swing.JLabel lblHastaCA;
    private javax.swing.JLabel lblHastaSC;
    private javax.swing.JMenuItem mitAcercaDe;
    private javax.swing.JMenuItem mitCambiarClave;
    private javax.swing.JMenuItem mitConsultarCampanias;
    private javax.swing.JMenuItem mitConsultarUsuarios;
    private javax.swing.JMenuItem mitCrearCampania;
    private javax.swing.JMenuItem mitCrearUsuario;
    private javax.swing.JMenuItem mitParametrosGen;
    private javax.swing.JMenuItem mitReporteBitacoraCampania;
    private javax.swing.JMenuItem mitReporteCostosCampanias;
```

```
    private javax.swing.JMenuItem mitReporteImpactoCampania;
    private javax.swing.JMenuItem mitReporteUsuarios;
    private javax.swing.JMenuItem mitSalir;
    private javax.swing.JMenu mnuAyuda;
    private javax.swing.JMenu mnuCampania;
    private javax.swing.JMenu mnuReportes;
    private javax.swing.JMenu mnuSistema;
    private javax.swing.JTable tabMonitor;
    private javax.swing.JTextField txtCampaniaCA;
    private javax.swing.JTextField txtCampaniaSC;
    private javax.swing.JTextField txtDesdeCA;
    private javax.swing.JTextField txtDesdeSC;
    private javax.swing.JTextField txtHastaCA;
    private javax.swing.JTextField txtHastaSC;
    // End of variables declaration
    private Usuario usuarioSesion;
    private Campania campaniaCA;
    private Vector<DetalleCampania> detallesCampania;
    private Campania campaniaSC;
    private Despachador procesoEnvio;
    private Vector<Tarea> monitor;
    private static Logger logger = Logger.getLogger(PantallaPrincipal.class);
}
```

4.1.11 Paquete raiz

4.1.11.1 Main

```
import com.jabluecatch.bo.ParametroGeneralBO;
import com.jabluecatch.entidades.ParametroGeneral;
import com.jabluecatch.gui.Login;
import com.jabluecatch.util.ErrorAplicativo;
import com.jabluecatch.util.Instalador;
import java.io.IOException;
import java.net.ServerSocket;
import javax.swing.JOptionPane;
import javax.swing.UIManager;
import org.apache.log4j.Logger;

/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author Fas
 */
public class Main {

    private static Logger logger = Logger.getLogger(Main.class);
```

```
    private static void setearApariencia() {

        //setear look and feel
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String args[]) {

        setearApariencia();

        //Verificar que no exista otra instancia de la aplicación
        try {
            ServerSocket ss = new ServerSocket(2806);
        } catch (IOException ioe) {
            logger.warn("La aplicación BlueCatch ya se está ejecutando. Error: " +
ioe.getMessage());
            JOptionPane.showMessageDialog(null, "La aplicación BlueCatch ya se está
ejecutando", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        //Verificar que BlueCatch se pueda comunicar con la base de datos
        boolean seDebeInstalar = false;
        ParametroGeneral pInstalado = null;
        try {
            pInstalado = ParametroGeneralBO.buscarPorParametro("INSTALADO");
        } catch (ErrorAplicativo ea) {
            if (!ea.getErrorTecnico().equals("Conexion.inicializaConexion - Unknown
database 'bluecatch'")) {
                logger.warn("Error de conexión a la base de datos. Error: " +
ea.getErrorTecnico());
                JOptionPane.showMessageDialog(null, "Error de conexión a la base de
datos", "Error", JOptionPane.ERROR_MESSAGE);
                return;
            } else {
                seDebeInstalar = true;
            }
        }

        if (pInstalado == null) {
            seDebeInstalar = true;
        } else if (!pInstalado.getValor().equals("S")) {
            seDebeInstalar = true;
        }

        if (seDebeInstalar) {
            int acepta = JOptionPane.showConfirmDialog(null, "El sistema no parece
estar instalado.\n "
                    + "Se procederá a instalar el mismo. Todos los datos se
reiniciarán. Desea continuar?", "Confirmación", JOptionPane.YES_NO_OPTION);
            if (acepta != 0) {
                return;
            }
```

```
            try {
                Instalador.ejecutar();
            } catch (ErrorAplicativo ea) {
                logger.warn("Error al instalar el sistema. Error: " +
ea.getErrorTecnico());
                JOptionPane.showMessageDialog(null, "Error al instalar el sistema",
"Error", JOptionPane.ERROR_MESSAGE);
                return;
            }
            JOptionPane.showMessageDialog(null, "El sistema se ha instalado
correctamente", "Información", JOptionPane.INFORMATION_MESSAGE);
        }

        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {
                new Login().setVisible(true);
            }
        });

    }
}
```

4.2 ARCHIVOS DE PROPIEDADES

4.2.1 Paquete raíz

4.2.1.1 log4j.properties

```
#Para dirigir mensajes a la salida estandar
log4j.appender.pantalla=org.apache.log4j.ConsoleAppender
log4j.appender.pantalla.Target=System.out
log4j.appender.pantalla.layout=org.apache.log4j.PatternLayout
log4j.appender.pantalla.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L - %m%n

#Para dirigir mensajes a un archivo
log4j.appender.archivo=org.apache.log4j.DailyRollingFileAppender
log4j.appender.archivo.DatePattern=dd-MM-yyyy'.'
log4j.appender.archivo.File=./logs/JABlueCatch.log
log4j.appender.archivo.layout=org.apache.log4j.PatternLayout
log4j.appender.archivo.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L - %m%n

#Root logger
log4j.rootLogger=info, archivo

#Configurar otros loggers
log4j.logger.com.jabluecatch=debug, pantalla
```

4.3 REPORTES JASPER

4.3.1 Paquete com.jabluecatch.recursos.reportes

4.3.1.1 ReporteBitacoraCampania.jrxml

```
<?xml version="1.0" encoding="UTF-8"?>
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd"
name="ReporteBitacoraCampania" language="groovy" pageWidth="842" pageHeight="595"
orientation="Landscape" columnWidth="802" leftMargin="20" rightMargin="20"
topMargin="20" bottomMargin="20">
	<property name="ireport.zoom" value="1.3310000000000084"/>
	<property name="ireport.x" value="0"/>
	<property name="ireport.y" value="0"/>
	<parameter name="P_ID_CAMPANIA" class="java.lang.Integer"
isForPrompting="false"/>
	<queryString>
		<![CDATA[SELECT c.campania,
	c.descripcion,
	dc.ruta,
	DATE_FORMAT(bc.fecha_registro,'%d/%m/%Y %H:%i:%S') fecha_registro,
	d.nombre_bt,
	d.direccion_bt,
	bc.observacion,
	u.usuario
FROM bitacora_campanias bc,
     detalle_campanias dc,
     campanias c,
     usuarios u,
     dispositivos d
WHERE dc.id_campania = $P{P_ID_CAMPANIA}
AND   dc.id_detalle_campania = bc.id_detalle_campania
AND   c.id_campania = dc.id_campania
AND   u.id_usuario = bc.id_usuario
AND   d.id_dispositivo = bc.id_dispositivo_destino
AND   bc.estado = 'A'
ORDER BY dc.id_detalle_campania, d.nombre_bt, bc.fecha_registro;]]>
	</queryString>
	<field name="campania" class="java.lang.String"/>
	<field name="descripcion" class="java.lang.String"/>
	<field name="ruta" class="java.lang.String"/>
	<field name="fecha_registro" class="java.lang.String"/>
	<field name="nombre_bt" class="java.lang.String"/>
	<field name="direccion_bt" class="java.lang.String"/>
	<field name="observacion" class="java.lang.String"/>
	<field name="usuario" class="java.lang.String"/>
	<group name="campania">
```

```
		<groupExpression><![CDATA[$F{campania}]]></groupExpression>
		<groupHeader>
			<band height="8">
				<line>
					<reportElement x="-20" y="-4" width="842"
height="1" forecolor="#666666"/>
				</line>
			</band>
		</groupHeader>
		<groupFooter>
			<band/>
		</groupFooter>
	</group>
	<group name="ruta">
		<groupExpression><![CDATA[$F{ruta}]]></groupExpression>
		<groupHeader>
			<band height="35">
				<textField>
					<reportElement x="100" y="0" width="722"
height="32" forecolor="#006699"/>
					<textElement>
						<font size="14" isBold="true"/>
					</textElement>
					<textFieldExpression
class="java.lang.String"><![CDATA[$F{ruta}]]></textFieldExpression>
				</textField>
				<staticText>
					<reportElement x="0" y="1" width="100"
height="31" forecolor="#666666"/>
					<textElement>
						<font size="12"/>
					</textElement>
					<text><![CDATA[Archivo de contenido:]]></text>
				</staticText>
			</band>
		</groupHeader>
		<groupFooter>
			<band/>
		</groupFooter>
	</group>
	<background>
		<band/>
	</background>
	<title>
		<band height="81">
			<frame>
				<reportElement mode="Opaque" x="-20" y="-20"
width="842" height="92" backcolor="#006699"/>
				<staticText>
					<reportElement x="20" y="20" width="385"
height="43" forecolor="#FFFFFF"/>
					<textElement>
						<font size="24" isBold="true"/>
					</textElement>
					<text><![CDATA[Reporte de bitácora de
Campaña]]></text>
				</staticText>
				<staticText>
```

```
                                        <reportElement x="552" y="20" width="270"
height="53" forecolor="#FFFFFF"/>
                                        <textElement textAlignment="Right"
verticalAlignment="Middle">
                                               <font size="14"/>
                                        </textElement>
                                        <text><![CDATA[Eventos generados durante la
transmisión de los archivos a los clientes]]></text>
                                </staticText>
                        </frame>
                </band>
        </title>
        <pageHeader>
                <band height="60">
                        <staticText>
                                <reportElement mode="Opaque" x="0" y="0" width="100"
height="28" forecolor="#666666" backcolor="#E6E6E6"/>
                                <textElement>
                                        <font size="12"/>
                                </textElement>
                                <text><![CDATA[Campaña]]></text>
                        </staticText>
                        <textField>
                                <reportElement mode="Opaque" x="100" y="0" width="702"
height="28" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement>
                                        <font size="18" isBold="true"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{campania}]]></textFieldExpression>
                        </textField>
                        <textField>
                                <reportElement mode="Opaque" x="0" y="28" width="802"
height="28" forecolor="#000000" backcolor="#E6E6E6"/>
                                <textElement>
                                        <font size="10" isBold="false"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{descripcion}]]></textFieldExpression>
                        </textField>
                </band>
        </pageHeader>
        <columnHeader>
                <band height="24">
                        <line>
                                <reportElement x="-20" y="-4" width="842" height="1"
forecolor="#666666"/>
                        </line>
                        <staticText>
                                <reportElement mode="Opaque" x="0" y="0" width="148"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Fecha Registro]]></text>
                        </staticText>
                        <staticText>
```

```
                                <reportElement mode="Opaque" x="148" y="0" width="149"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Nombre Dispositivo]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="297" y="0" width="121"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Dirección]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="691" y="0" width="111"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Usuario]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="418" y="1" width="273"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Observación]]></text>
                        </staticText>
                </band>
        </columnHeader>
        <detail>
                <band height="20">
                        <line>
                                <reportElement positionType="FixRelativeToBottom" x="0"
y="18" width="822" height="1"/>
                        </line>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="0" y="0" width="148" height="18"/>
                                <textElement>
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{fecha_registro}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="148" y="0" width="149" height="18"/>
                                <textElement>
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{nombre_bt}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="297" y="0" width="121" height="18"/>
                                <textElement>
```

```
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{direccion_bt}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                            <reportElement x="418" y="0" width="273" height="18"/>
                            <textElement>
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{observacion}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                            <reportElement x="691" y="0" width="111" height="18"/>
                            <textElement>
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{usuario}]]></textFieldExpression>
                        </textField>
                </band>
        </detail>
        <columnFooter>
                <band/>
        </columnFooter>
        <pageFooter>
                <band height="20">
                        <textField>
                            <reportElement mode="Opaque" x="0" y="4" width="762"
height="13" backcolor="#E6E6E6"/>
                            <textElement textAlignment="Right"/>
                            <textFieldExpression
class="java.lang.String"><![CDATA["Página "+$V{PAGE_NUMBER}+"
de"]]></textFieldExpression>
                        </textField>
                        <textField evaluationTime="Report">
                            <reportElement mode="Opaque" x="762" y="4" width="40"
height="13" backcolor="#E6E6E6"/>
                            <textElement/>
                            <textFieldExpression
class="java.lang.String"><![CDATA[" " + $V{PAGE_NUMBER}]]></textFieldExpression>
                        </textField>
                        <textField pattern="EEEEE dd MMMMM yyyy">
                            <reportElement x="0" y="4" width="100" height="13"/>
                            <textElement/>
                            <textFieldExpression
class="java.util.Date"><![CDATA[new java.util.Date()]]></textFieldExpression>
                        </textField>
                </band>
        </pageFooter>
        <summary>
                <band/>
        </summary>
</jasperReport>
```

4.3.1.2 ReporteCostosCampanias.jrxml

```
<?xml version="1.0" encoding="UTF-8"?>
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd"
name="ReporteCostosCampanias" language="groovy" pageWidth="842" pageHeight="595"
orientation="Landscape" columnWidth="802" leftMargin="20" rightMargin="20"
topMargin="20" bottomMargin="20">
	<property name="ireport.zoom" value="1.4641000000000022"/>
	<property name="ireport.x" value="0"/>
	<property name="ireport.y" value="88"/>
	<parameter name="P_FECHA_INICIO" class="java.sql.Timestamp"
isForPrompting="false"/>
	<parameter name="P_FECHA_FIN" class="java.sql.Timestamp"
isForPrompting="false"/>
	<queryString>
		<![CDATA[SELECT

  c.campania,
  c.descripcion,
  c.fecha_inicio f_fecha_inicio,

  DATE_FORMAT(c.fecha_inicio,'%d/%m/%Y %H:%i:%S') s_fecha_inicio,
  DATE_FORMAT(c.fecha_fin,'%d/%m/%Y %H:%i:%S') s_fecha_fin,

  (SELECT IFNULL(COUNT(*),0) CANTIDAD
  FROM bitacora_campanias bc2,
       detalle_campanias dc2,
       campanias c2
  WHERE dc2.id_detalle_campania = bc2.id_detalle_campania
  AND   dc2.id_campania = c.id_campania
  AND   c2.id_campania = dc2.id_campania
  AND   bc2.evento = 'E'
  AND   bc2.estado = 'A') MENSAJES_ENVIADOS,

  LPAD(FORMAT(c.costo_ref_mensaje, 2), 4, '0') S_COSTO_MENSAJE,

  lpad(format(((SELECT IFNULL(COUNT(*),0)
  FROM bitacora_campanias bc2,
       detalle_campanias dc2,
       campanias c2
  WHERE dc2.id_detalle_campania = bc2.id_detalle_campania
  AND   dc2.id_campania = c.id_campania
  AND   c2.id_campania = dc2.id_campania
  AND   bc2.evento = 'E'
  AND   bc2.estado = 'A') * c.costo_ref_mensaje), 2), 4, '0') TOTAL,

  ((SELECT IFNULL(COUNT(*),0)
  FROM bitacora_campanias bc2,
       detalle_campanias dc2,
       campanias c2
  WHERE dc2.id_detalle_campania = bc2.id_detalle_campania
  AND   dc2.id_campania = c.id_campania
  AND   c2.id_campania = dc2.id_campania
```

```
  AND   bc2.evento = 'E'
  AND   bc2.estado = 'A') * c.costo_ref_mensaje) TOTAL_NUM


FROM campanias c
WHERE c.estado = 'A'
AND   c.fecha_inicio > $P{P_FECHA_INICIO}
AND   IFNULL(c.fecha_fin,now()) <= $P{P_FECHA_FIN}
ORDER BY c.fecha_inicio ASC;]]>
       </queryString>
       <field name="campania" class="java.lang.String"/>
       <field name="descripcion" class="java.lang.String"/>
       <field name="f_fecha_inicio" class="java.sql.Timestamp"/>
       <field name="s_fecha_inicio" class="java.lang.String"/>
       <field name="s_fecha_fin" class="java.lang.String"/>
       <field name="MENSAJES_ENVIADOS" class="java.lang.Long"/>
       <field name="S_COSTO_MENSAJE" class="java.lang.String"/>
       <field name="TOTAL" class="java.lang.String"/>
       <field name="TOTAL_NUM" class="java.lang.Double"/>
       <background>
              <band/>
       </background>
       <title>
              <band height="72">
                     <frame>
                            <reportElement mode="Opaque" x="-20" y="-20"
width="842" height="92" backcolor="#006699"/>
                            <staticText>
                                   <reportElement x="20" y="20" width="399"
height="43" forecolor="#FFFFFF"/>
                                   <textElement>
                                          <font size="24" isBold="true"/>
                                   </textElement>
                                   <text><![CDATA[Reporte de Costos de
Campañas]]></text>
                            </staticText>
                            <staticText>
                                   <reportElement x="642" y="20" width="180"
height="43" forecolor="#FFFFFF"/>
                                   <textElement textAlignment="Right">
                                          <font size="14" isBold="false"/>
                                   </textElement>
                                   <text><![CDATA[Reporte histórico de los costos
de las campañas]]></text>
                            </staticText>
                     </frame>
              </band>
       </title>
       <pageHeader>
              <band height="13"/>
       </pageHeader>
       <columnHeader>
              <band height="56">
                     <line>
                            <reportElement x="-20" y="44" width="822" height="1"
forecolor="#666666"/>
                     </line>
                     <staticText>
```

```
                        <reportElement mode="Opaque" x="0" y="0" width="150"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Campaña]]></text>
                </staticText>
                <staticText>
                        <reportElement mode="Opaque" x="314" y="0" width="150"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Fecha de inicio]]></text>
                </staticText>
                <staticText>
                        <reportElement mode="Opaque" x="464" y="0" width="150"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Fecha de fin]]></text>
                </staticText>
                <staticText>
                        <reportElement mode="Opaque" x="614" y="0" width="62"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Mensajes
Enviados]]></text>
                </staticText>
                <staticText>
                        <reportElement mode="Opaque" x="676" y="0" width="56"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Costo
Mensaje]]></text>
                </staticText>
                <staticText>
                        <reportElement mode="Opaque" x="732" y="0" width="70"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Total]]></text>
                </staticText>
                <staticText>
                        <reportElement mode="Opaque" x="150" y="0" width="164"
height="40" forecolor="#006699" backcolor="#E6E6E6"/>
```

```
                                <textElement textAlignment="Center"
verticalAlignment="Middle">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Descripción de la
Campaña]]></text>
                        </staticText>
                </band>
        </columnHeader>
        <detail>
                <band height="21">
                        <line>
                                <reportElement positionType="FixRelativeToBottom" x="0"
y="19" width="802" height="1"/>
                        </line>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="0" y="0" width="150" height="18"/>
                                <textElement>
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{campania}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="314" y="0" width="150" height="18"/>
                                <textElement>
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{s_fecha_inicio}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="464" y="0" width="150" height="18"/>
                                <textElement>
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{s_fecha_fin}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="614" y="0" width="62" height="18"/>
                                <textElement textAlignment="Center">
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.Long"><![CDATA[$F{MENSAJES_ENVIADOS}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="676" y="0" width="56" height="18"/>
                                <textElement textAlignment="Right"
verticalAlignment="Middle">
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA["\$ " +
$F{S_COSTO_MENSAJE}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
```

```
                                <reportElement x="732" y="0" width="70" height="18"/>
                                <textElement textAlignment="Right"
verticalAlignment="Middle">
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA["\$ " + $F{TOTAL}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="150" y="1" width="164" height="18"/>
                                <textElement>
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{descripcion}]]></textFieldExpression>
                        </textField>
                </band>
        </detail>
        <columnFooter>
                <band/>
        </columnFooter>
        <pageFooter>
                <band height="17">
                        <textField>
                                <reportElement mode="Opaque" x="0" y="4" width="762"
height="13" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Right"/>
                                <textFieldExpression
class="java.lang.String"><![CDATA["Página "+$V{PAGE_NUMBER}+"
de"]]></textFieldExpression>
                        </textField>
                        <textField evaluationTime="Report">
                                <reportElement mode="Opaque" x="762" y="4" width="40"
height="13" backcolor="#E6E6E6"/>
                                <textElement/>
                                <textFieldExpression
class="java.lang.String"><![CDATA[" " + $V{PAGE_NUMBER}]]></textFieldExpression>
                        </textField>
                        <textField pattern="EEEEE dd MMMMM yyyy">
                                <reportElement x="0" y="4" width="100" height="13"/>
                                <textElement/>
                                <textFieldExpression
class="java.util.Date"><![CDATA[new java.util.Date()]]></textFieldExpression>
                        </textField>
                </band>
        </pageFooter>
        <summary>
                <band height="292">
                        <timeSeriesChart>
                                <chart theme="eye.candy.sixties">
                                        <reportElement x="13" y="52" width="775"
height="229"/>
                                        <chartTitle/>
                                        <chartSubtitle/>
                                        <chartLegend/>
                                </chart>
                                <timeSeriesDataset timePeriod="Minute">
                                        <timeSeries>
```

```
                                        <seriesExpression><![CDATA["Costos de
Campañas"]]></seriesExpression>

        <timePeriodExpression><![CDATA[$F{f_fecha_inicio}]]></timePeriodExpression>

        <valueExpression><![CDATA[$F{TOTAL_NUM}]]></valueExpression>
                                </timeSeries>
                        </timeSeriesDataset>
                        <timeSeriesPlot>
                                <plot/>
                                <timeAxisFormat>
                                        <axisFormat>
                                                <labelFont/>
                                                <tickLabelFont/>
                                        </axisFormat>
                                </timeAxisFormat>
                                <valueAxisFormat>
                                        <axisFormat>
                                                <labelFont/>
                                                <tickLabelFont/>
                                        </axisFormat>
                                </valueAxisFormat>
                        </timeSeriesPlot>
                </timeSeriesChart>
        </band>
        </summary>
</jasperReport>
```

4.3.1.3 ReporteImpactoCampania.jrxml

```
<?xml version="1.0" encoding="UTF-8"?>
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd"
name="ReporteImpactoCampania" language="groovy" pageWidth="595" pageHeight="842"
columnWidth="535" leftMargin="20" rightMargin="20" topMargin="20"
bottomMargin="20">
        <property name="ireport.zoom" value="1.000000000000001"/>
        <property name="ireport.x" value="0"/>
        <property name="ireport.y" value="19"/>
        <parameter name="P_ID_CAMPANIA" class="java.lang.Integer"
isForPrompting="false"/>
        <queryString>
                <![CDATA[SELECT c.campania, c.descripcion, 'Transferencias exitosas'
GRUPO, IFNULL(COUNT(*),0) CANTIDAD
FROM bitacora_campanias bc,
     detalle_campanias dc,
     campanias c
WHERE dc.id_detalle_campania = bc.id_detalle_campania
AND   dc.id_campania = $P{P_ID_CAMPANIA}
AND   c.id_campania = dc.id_campania
AND   bc.evento = 'E'
AND   bc.estado = 'A'
UNION
```

```
SELECT c.campania, c.descripcion, 'Transferencias canceladas' GRUPO,
IFNULL(COUNT(*),0) CANTIDAD
FROM bitacora_campanias bc,
     detalle_campanias dc,
     campanias c
WHERE dc.id_detalle_campania = bc.id_detalle_campania
AND   dc.id_campania = $P{P_ID_CAMPANIA}
AND   c.id_campania = dc.id_campania
AND   bc.evento = 'C'
AND   bc.estado = 'A'
UNION
SELECT c.campania, c.descripcion, 'Transferencias fallidas' GRUPO,
IFNULL(COUNT(*),0) CANTIDAD
FROM bitacora_campanias bc,
     detalle_campanias dc,
     campanias c
WHERE dc.id_detalle_campania = bc.id_detalle_campania
AND   dc.id_campania = $P{P_ID_CAMPANIA}
AND   c.id_campania = dc.id_campania
AND   bc.evento = 'X'
AND   bc.observacion NOT LIKE '%No se ha encontrado el servicio%'
AND   bc.estado = 'A';]]>
       </queryString>
       <field name="campania" class="java.lang.String"/>
       <field name="descripcion" class="java.lang.String"/>
       <field name="GRUPO" class="java.lang.String"/>
       <field name="CANTIDAD" class="java.lang.Long"/>
       <group name="campania">
              <groupExpression><![CDATA[$F{campania}]]></groupExpression>
              <groupHeader>
                     <band height="9">
                            <line>
                                   <reportElement x="-20" y="-28" width="595"
height="1" forecolor="#666666"/>
                            </line>
                     </band>
              </groupHeader>
              <groupFooter>
                     <band/>
              </groupFooter>
       </group>
       <background>
              <band/>
       </background>
       <title>
              <band height="129">
                     <frame>
                            <reportElement mode="Opaque" x="-20" y="-20"
width="595" height="111" backcolor="#006699"/>
                            <staticText>
                                   <reportElement x="20" y="20" width="555"
height="43" forecolor="#FFFFFF"/>
                                   <textElement>
                                          <font size="24" isBold="true"/>
                                   </textElement>
                                   <text><![CDATA[Reporte de Impacto de
Campaña]]></text>
                            </staticText>
```

```
                        <staticText>
                                <reportElement x="20" y="63" width="554"
height="48" forecolor="#FFFFFF"/>
                                <textElement>
                                        <font size="14" isBold="false"/>
                                </textElement>
                                <text><![CDATA[Impacto que ha tenido la campaña
de acuerdo a las cantidades de mensajes aceptados, rechazados y
cancelados]]></text>
                        </staticText>
                </frame>
        </band>
    </title>
    <pageHeader>
        <band height="81">
                <staticText>
                        <reportElement mode="Opaque" x="0" y="1" width="99"
height="28" forecolor="#666666" backcolor="#E6E6E6"/>
                        <textElement>
                                <font size="12"/>
                        </textElement>
                        <text><![CDATA[Campaña]]></text>
                </staticText>
                <textField>
                        <reportElement mode="Opaque" x="99" y="1" width="456"
height="28" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement>
                                <font size="18" isBold="true"/>
                        </textElement>
                        <textFieldExpression
class="java.lang.String"><![CDATA[$F{campania}]]></textFieldExpression>
                </textField>
                <textField>
                        <reportElement mode="Opaque" x="0" y="29" width="555"
height="48" forecolor="#000000" backcolor="#E6E6E6"/>
                        <textElement>
                                <font size="10" isBold="false"/>
                        </textElement>
                        <textFieldExpression
class="java.lang.String"><![CDATA[$F{descripcion}]]></textFieldExpression>
                </textField>
        </band>
    </pageHeader>
    <columnHeader>
        <band height="24">
                <line>
                        <reportElement x="-20" y="20" width="595" height="1"
forecolor="#666666"/>
                </line>
                <staticText>
                        <reportElement mode="Opaque" x="0" y="0" width="277"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Grupo]]></text>
                </staticText>
                <staticText>
```

```
                        <reportElement mode="Opaque" x="277" y="0" width="277"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Center">
                                <font size="12" isBold="true"/>
                        </textElement>
                        <text><![CDATA[Cantidad]]></text>
                </staticText>
        </band>
</columnHeader>
<detail>
        <band height="21">
                <line>
                        <reportElement positionType="FixRelativeToBottom" x="0"
y="18" width="555" height="1"/>
                </line>
                <textField isStretchWithOverflow="true">
                        <reportElement x="0" y="0" width="277" height="18"/>
                        <textElement textAlignment="Center">
                                <font size="10"/>
                        </textElement>
                        <textFieldExpression
class="java.lang.String"><![CDATA[$F{GRUPO}]]></textFieldExpression>
                </textField>
                <textField isStretchWithOverflow="true">
                        <reportElement x="277" y="0" width="277" height="18"/>
                        <textElement textAlignment="Center">
                                <font size="10"/>
                        </textElement>
                        <textFieldExpression
class="java.lang.Long"><![CDATA[$F{CANTIDAD}]]></textFieldExpression>
                </textField>
        </band>
</detail>
<columnFooter>
        <band/>
</columnFooter>
<pageFooter>
        <band height="20">
                <textField>
                        <reportElement mode="Opaque" x="0" y="4" width="515"
height="13" backcolor="#E6E6E6"/>
                        <textElement textAlignment="Right"/>
                        <textFieldExpression
class="java.lang.String"><![CDATA["Página "+$V{PAGE_NUMBER}+"
de"]]></textFieldExpression>
                </textField>
                <textField evaluationTime="Report">
                        <reportElement mode="Opaque" x="515" y="4" width="40"
height="13" backcolor="#E6E6E6"/>
                        <textElement/>
                        <textFieldExpression
class="java.lang.String"><![CDATA[" " + $V{PAGE_NUMBER}]]></textFieldExpression>
                </textField>
                <textField pattern="EEEEE dd MMMMM yyyy">
                        <reportElement x="0" y="4" width="100" height="13"/>
                        <textElement/>
                        <textFieldExpression
class="java.util.Date"><![CDATA[new java.util.Date()]]></textFieldExpression>
```

```
                </textField>
            </band>
        </pageFooter>
        <summary>
            <band height="303">
                <pie3DChart>
                    <chart theme="default">
                        <reportElement x="22" y="44" width="514"
height="245"/>
                        <chartTitle>
                            <titleExpression><![CDATA["Impacto de la
Campaña: "+$F{campania}]]></titleExpression>
                        </chartTitle>
                        <chartSubtitle/>
                        <chartLegend/>
                    </chart>
                    <pieDataset>

        <keyExpression><![CDATA[$F{GRUPO}]]></keyExpression>

        <valueExpression><![CDATA[$F{CANTIDAD}]]></valueExpression>
                    </pieDataset>
                    <pie3DPlot isCircular="false">
                        <plot/>
                        <itemLabel color="#000000"
backgroundColor="#FFFFFF"/>
                    </pie3DPlot>
                </pie3DChart>
            </band>
        </summary>
</jasperReport>
```

4.3.1.4 ReporteUsuarios.jrxml

```
<?xml version="1.0" encoding="UTF-8"?>
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd" name="ReporteUsuarios"
language="groovy" pageWidth="842" pageHeight="595" orientation="Landscape"
columnWidth="802" leftMargin="20" rightMargin="20" topMargin="20"
bottomMargin="20">
        <property name="ireport.zoom" value="1.1000000000000016"/>
        <property name="ireport.x" value="0"/>
        <property name="ireport.y" value="0"/>
        <queryString language="SQL">
            <![CDATA[SELECT
     usuarios.`id_usuario` AS usuarios_id_usuario,
     usuarios.`usuario` AS usuarios_usuario,
     roles.`rol` AS roles_rol,
     usuarios.`nombres` AS usuarios_nombres,
     usuarios.`apellidos` AS usuarios_apellidos,
     usuarios.`identificacion` AS usuarios_identificacion
FROM
```

```
     `roles` roles INNER JOIN `usuarios` usuarios ON roles.`id_rol` =
usuarios.`id_rol`
WHERE
     usuarios.estado = 'A'
 AND roles.estado = 'A']]>
       </queryString>
       <field name="usuarios_id_usuario" class="java.lang.Integer"/>
       <field name="usuarios_usuario" class="java.lang.String"/>
       <field name="roles_rol" class="java.lang.String"/>
       <field name="usuarios_nombres" class="java.lang.String"/>
       <field name="usuarios_apellidos" class="java.lang.String"/>
       <field name="usuarios_identificacion" class="java.lang.String"/>
       <background>
              <band/>
       </background>
       <title>
              <band height="56">
                     <frame>
                            <reportElement mode="Opaque" x="-20" y="-20"
width="842" height="75" backcolor="#006699"/>
                            <staticText>
                                   <reportElement x="20" y="21" width="332"
height="43" forecolor="#FFFFFF"/>
                                   <textElement>
                                          <font size="24" isBold="true"/>
                                   </textElement>
                                   <text><![CDATA[Reporte de usuarios]]></text>
                            </staticText>
                            <staticText>
                                   <reportElement x="596" y="27" width="226"
height="37" forecolor="#FFFFFF"/>
                                   <textElement textAlignment="Right">
                                          <font size="14" isBold="false"/>
                                   </textElement>
                                   <text><![CDATA[Listado de los usuarios del
sistema]]></text>
                            </staticText>
                     </frame>
              </band>
       </title>
       <pageHeader>
              <band height="10"/>
       </pageHeader>
       <columnHeader>
              <band height="21">
                     <line>
                            <reportElement x="-20" y="20" width="822" height="1"
forecolor="#666666"/>
                     </line>
                     <staticText>
                            <reportElement mode="Opaque" x="0" y="0" width="92"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                            <textElement textAlignment="Center">
                                   <font size="12" isBold="true"/>
                            </textElement>
                            <text><![CDATA[Id]]></text>
                     </staticText>
                     <staticText>
```

```
                                <reportElement mode="Opaque" x="92" y="0" width="111"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Usuario]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="203" y="0" width="110"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Rol]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="313" y="0" width="180"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Nombres]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="493" y="0" width="180"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Apellidos]]></text>
                        </staticText>
                        <staticText>
                                <reportElement mode="Opaque" x="673" y="0" width="129"
height="20" forecolor="#006699" backcolor="#E6E6E6"/>
                                <textElement textAlignment="Center">
                                        <font size="12" isBold="true"/>
                                </textElement>
                                <text><![CDATA[Identificación]]></text>
                        </staticText>
                </band>
        </columnHeader>
        <detail>
                <band height="20">
                        <line>
                                <reportElement positionType="FixRelativeToBottom" x="0"
y="17" width="802" height="1"/>
                        </line>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="0" y="0" width="92" height="18"/>
                                <textElement textAlignment="Center">
                                        <font size="10"/>
                                </textElement>
                                <textFieldExpression
class="java.lang.Integer"><![CDATA[$F{usuarios_id_usuario}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                                <reportElement x="92" y="0" width="111" height="18"/>
                                <textElement>
```

```
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{usuarios_usuario}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                            <reportElement x="203" y="0" width="110" height="18"/>
                            <textElement>
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{roles_rol}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                            <reportElement x="313" y="0" width="180" height="18"/>
                            <textElement>
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{usuarios_nombres}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                            <reportElement x="493" y="0" width="180" height="18"/>
                            <textElement>
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{usuarios_apellidos}]]></textFieldExpression>
                        </textField>
                        <textField isStretchWithOverflow="true">
                            <reportElement x="673" y="0" width="129" height="18"/>
                            <textElement>
                                <font size="10"/>
                            </textElement>
                            <textFieldExpression
class="java.lang.String"><![CDATA[$F{usuarios_identificacion}]]></textFieldExpressi
on>
                        </textField>
                </band>
        </detail>
        <columnFooter>
                <band/>
        </columnFooter>
        <pageFooter>
                <band height="17">
                        <textField>
                            <reportElement mode="Opaque" x="0" y="4" width="762"
height="13" backcolor="#E6E6E6"/>
                            <textElement textAlignment="Right"/>
                            <textFieldExpression
class="java.lang.String"><![CDATA["Página "+$V{PAGE_NUMBER}+"
de"]]></textFieldExpression>
                        </textField>
                        <textField evaluationTime="Report">
                            <reportElement mode="Opaque" x="762" y="4" width="40"
height="13" backcolor="#E6E6E6"/>
                            <textElement/>
```

```
                        <textFieldExpression
class="java.lang.String"><![CDATA[" " + $V{PAGE_NUMBER}]]></textFieldExpression>
                    </textField>
                    <textField pattern="EEEEE dd MMMMM yyyy">
                        <reportElement x="0" y="4" width="100" height="13"/>
                        <textElement/>
                        <textFieldExpression
class="java.util.Date"><![CDATA[new java.util.Date()]]></textFieldExpression>
                    </textField>
            </band>
        </pageFooter>
        <summary>
            <band/>
        </summary>
</jasperReport>
```

UNIVERSIDAD DE GUAYAQUIL

FACULTAD DE CIENCIAS MATEMATICAS Y FISICAS

CARRERA DE INGENIERIA EN SISTEMAS COMPUTACIONALES

IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE ENVÍO DE PUBLICIDAD VÍA BLUETOOTH PARA UN ALMACÉN DE MÚSICA, COMO APLICACIÓN DEL CONCEPTO DE MARKETING DE PROXIMIDAD

MANUAL DE USUARIO

AUTOR: FAUSTO STANLEY ALMEIDA CAMPOS

TUTOR: ING. DAVID BENAVIDES

GUAYAQUIL – ECUADOR

2010

Guayaquil, 16 de agosto del 2010

1 INSTALACIÓN

1.1 REQUERIMIENTOS RECOMENDADOS DE HARDWARE

Si el computador solamente se encargará de ejecutar y mantener en línea el sistema BlueCatch, se recomiendan los siguiente requerimientos de hardware:

- Procesador: Pentium Dual Core 2 GHz o equivalente.
- Memoria RAM: 1 Gb.
- Disco duro: 40 Mb (sólo el software[1]).
- Dispositivo transmisor de datos vía Bluetooth (USB)

1.2 REQUERIMIENTOS DE SOFTWARE

El computador deberá poseer los siguientes requisitos en cuanto a software se refiere:

- Sistema Operativo: Microsoft Windows XP Service Pack 3.
- Microsoft Bluetooth Stack (o algún software similar de terceros).
- Java Runtime Enviroment v.1.6 update 20 o superior.
- Base de datos MySQL v.5.1.46 o superior. Con una cuenta *root* y contraseña *mysql*.
- Visor de archivos PDF. Por ejemplo: Adobe Reader.

Los pasos de instalación del Java Runtime Enviroment y la base de datos MySQL se detallan como anexos a este documento.

[1] Este espacio es necesario para la instalación del software únicamente. El espacio en disco duro necesario para los datos que genere el sistema por la ejecución de las campañas se detalla en al Manual Técnico del sistema.

1.3 PASOS DE INSTALACIÓN

Para poder instalar el software solo es necesario seguir los siguientes pasos:

- Copiar la carpeta *BlueCatch* en el directorio raíz: C:\.
- Crear un acceso directo del archivo *JABlueCatch.jar* en el escritorio.
- Ejecutar desde el acceso directo creado.

2 ORGANIZACIÓN DEL MENU

2.1 ROL ADMINISTRADOR

2.1.1 Sistema

Menú que contiene las principales funciones administrativas del sistema.

- Crear usuario
- Consultar usuarios
- Parámetros generales

2.1.2 Campaña

Menú que contiene las principales funciones para administrar todas las campañas publicitarias que se ejecuten desde el sistema.

- Crear campaña
- Consultar campañas

2.1.3 Reportes

Menú que contiene las opciones necesarias para ejecutar los reportes administrativos y operativos del sistema.

- Reporte de usuarios
- Reporte de Bitácora por Campaña

2.2 ROL OPERADOR

2.2.1 Sistema

Menú que contiene la función de cambio de clave del usuario respectivo.

- Cambiar clave
- Salir

2.2.2 Campaña

Menú que contiene las principales funciones para consultar todas las campañas publicitarias que se ejecuten desde el sistema.

- Consultar campañas

2.2.3 Reportes

Menú que contiene las opciones necesarias para ejecutar los reportes operativos del sistema.

- Reporte de Impacto de Campaña
- Reporte de Costos de Campañas

3 OPCIONES DEL SISTEMA

3.1 INICIO DEL SISTEMA

3.1.1 Login

Esta es la primera pantalla en aparecer. Sirve para ingresar las credenciales de inicio de sesión de cada usuario. Dependiendo del rol que posea el usuario con el que se inicia la sesión en el sistema, se activarán las opciones dentro del sistema.

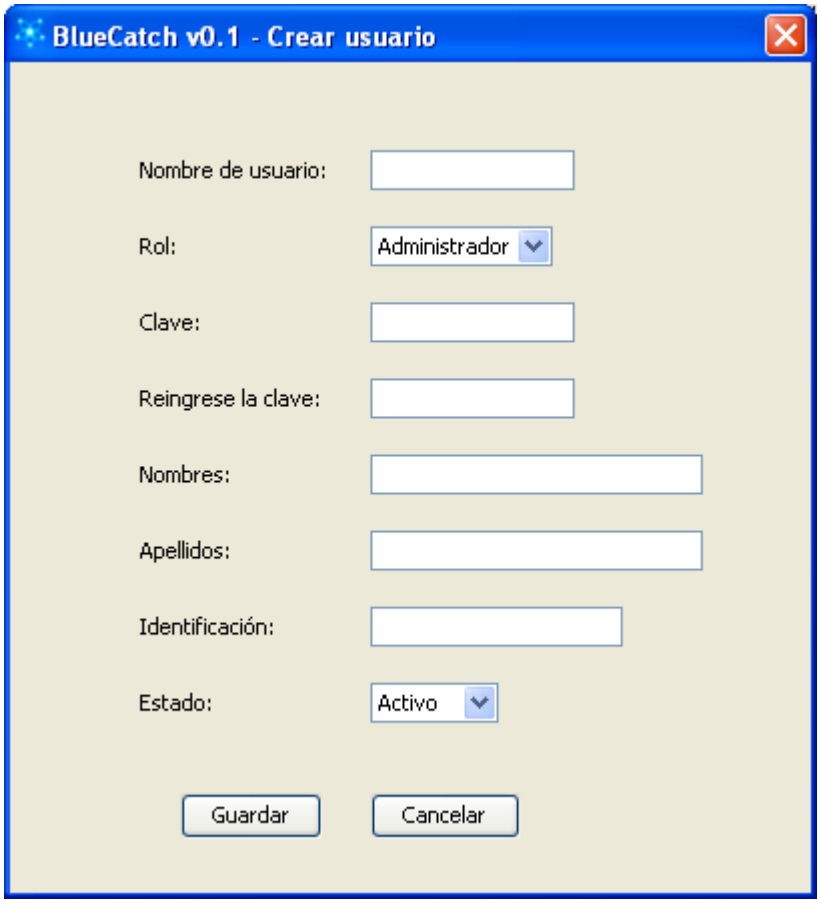
Nombre de Campo	**Descripción del Campo**
Usuario	Nombre de usuario.
Clave	Contraseña respectiva al usuario.
Botón Aceptar	Inicia la sesión del sistema.
Botón Cancelar	Cancela el inicio de sesión y finaliza el sistema.
Características	
Se debe ingresar un usuario y una clave previamente configurados en el sistema.	

3.2 ROL ADMINISTRADOR

3.2.1 Menú: Sistema

3.2.1.1 Opción: Crear usuario

Esta opción dentro del menú *Sistema* permite al usuario administrador del sistema crear nuevos usuarios administradores u operadores del sistema, con sus respectivos datos personales.

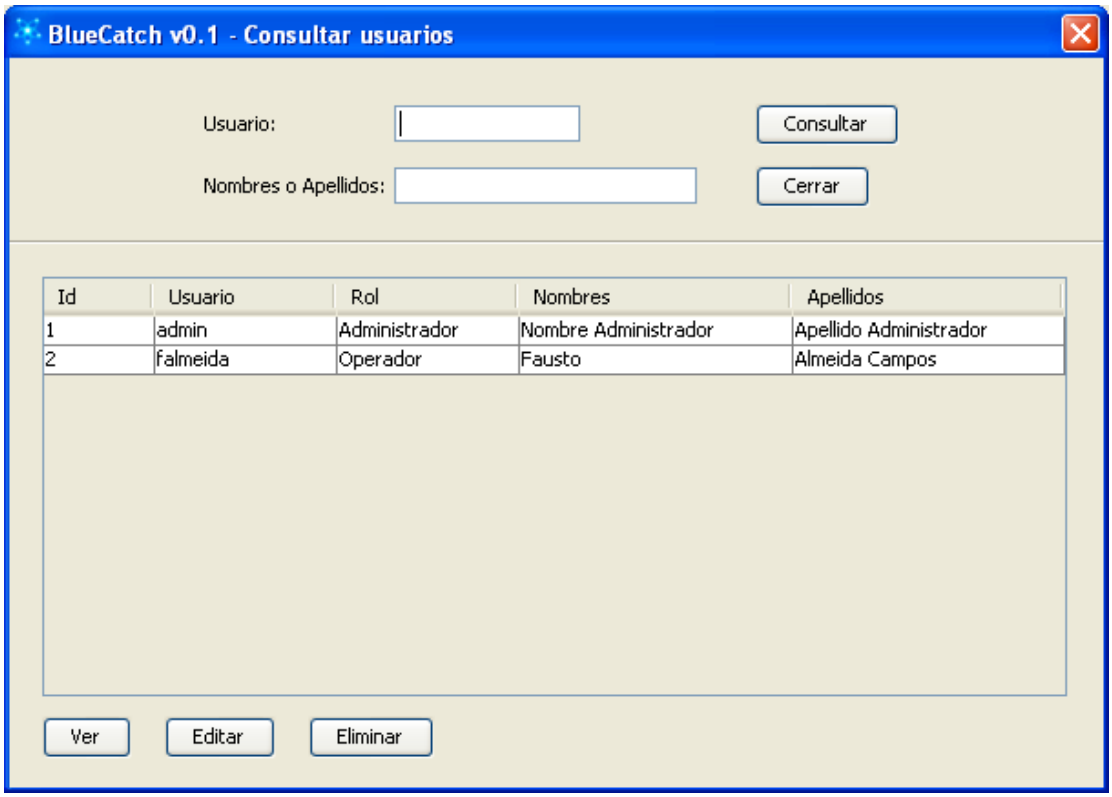


Nombre de Campo	Descripción del Campo
Nombre de usuario	Nombre de usuario. Sólo se permiten letras en minúscula y dígitos del 0 al 9.
Rol	Rol que va a poseer el usuario que se está creando.
Clave	Contraseña respectiva al usuario.
Reingrese la clave	Aquí se vuelve a escribir la contraseña respectiva a manera de confirmación.
Nombres	Nombres del usuario respectivo.

Apellidos	Apellidos del usuario respectivo.
Identificación	Número de cédula o pasaporte del usuario respectivo.
Estado	Estado del usuario.
Botón Guardar	Realiza las validaciones necesarias y guarda el nuevo usuario en la base de datos.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.
Características	
Un usuario creado como inactivo no podrá iniciar sesión en el sistema mientras el administrador no lo active explícitamente.	

3.2.1.2 Opción: Consultar usuarios

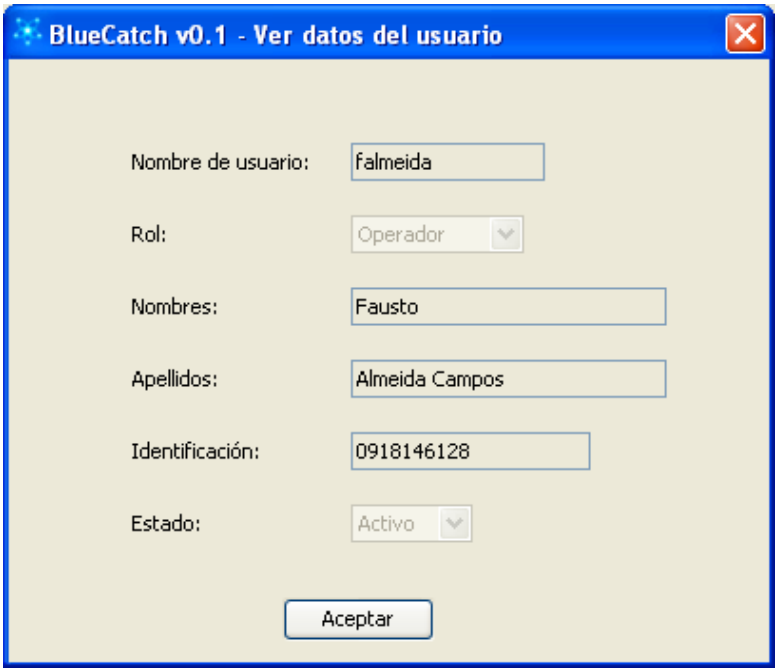
Esta opción dentro del menú *Sistema* permite al usuario administrador del sistema consultar lo usuarios que se encuentran creados en el sistema.



Nombre de Campo	Descripción del Campo
Usuario	Nombre de usuario completo o fracción del mismo. Se utiliza para realizar la búsqueda por un patrón.
Nombres o Apellidos	Nombres o apellidos completos del usuario o fracciones de los mismos. Se utilizan para realizar la búsqueda por un patrón.
Id	Código del usuario.
Usuario	Nombre de usuario.
Rol	Rol que posee el usuario.
Nombres	Nombres del usuario respectivo.
Apellidos	Apellidos del usuario respectivo.
Botón Consultar	Realiza la consulta de los usuarios de acuerdo a los criterios ingresados.
Botón Cerrar	Cierra la ventana y descarta cualquier cambio realizado.
Botón Ver	Abre una ventana que permite visualizar los datos del usuario seleccionado.
Botón Editar	Abre una ventana que permite editar los datos del usuario seleccionado.
Botón Eliminar	Permite eliminar el usuario del sistema, previa confirmación.
Características	
El campo del formulario de criterios de búsqueda que quede vacío no se tomará en cuenta para realizar la consulta.	

3.2.1.2.1 Opción: Ver datos del usuario

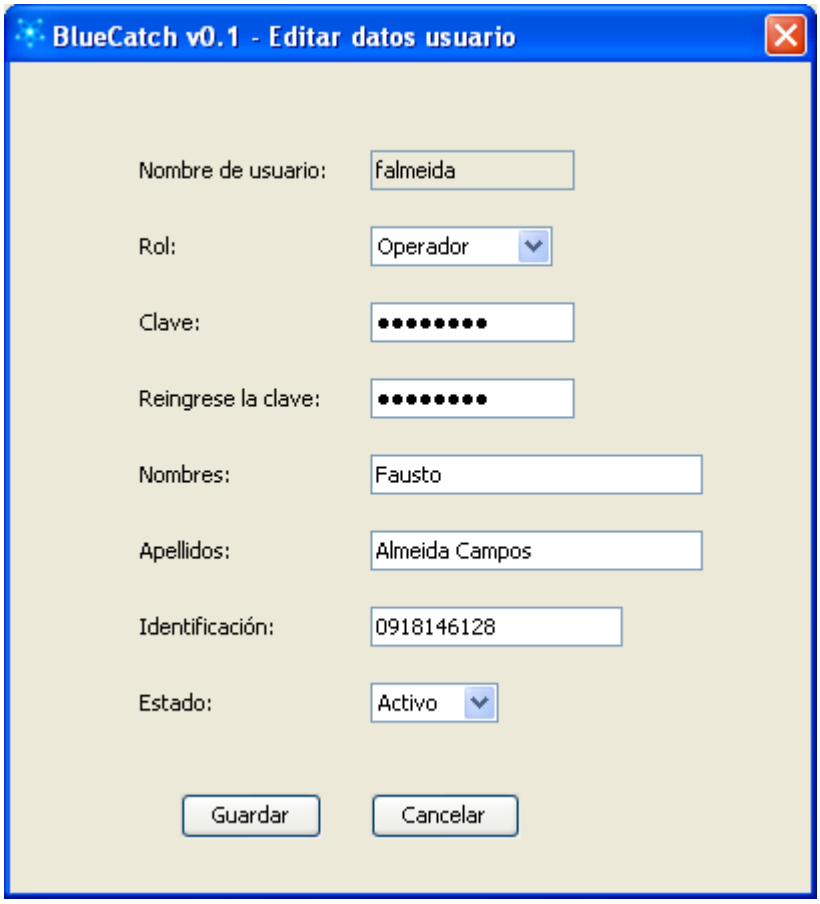
Permite visualizar el detalle de los datos del usuario seleccionado.



Nombre de Campo	**Descripción del Campo**
Nombre de usuario	Nombre de usuario. Sólo se permiten letras en minúscula y dígitos del 0 al 9.
Rol	Rol que va a poseer el usuario que se está creando.
Nombres	Nombres del usuario respectivo.
Apellidos	Apellidos del usuario respectivo.
Identificación	Número de cédula o pasaporte del usuario respectivo.
Estado	Estado del usuario.
Características	
No aplica.	

3.2.1.2.2 Opción: Editar datos del usuario

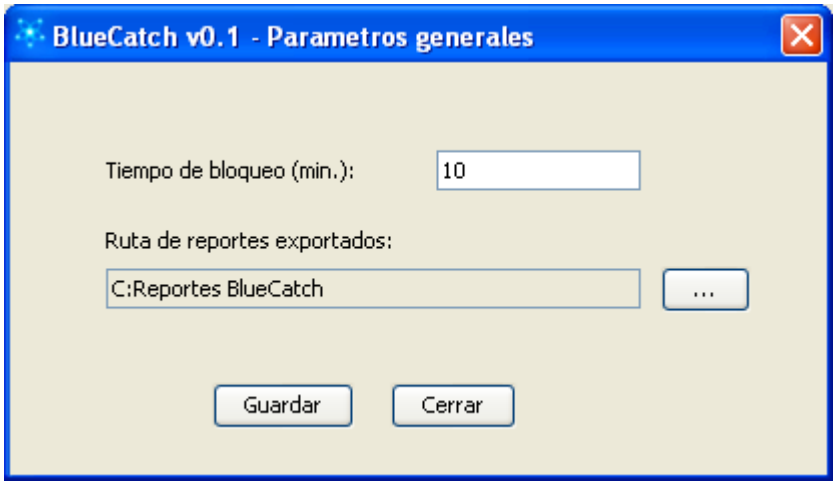
Permite editar el detalle de los datos del usuario seleccionado.



Nombre de Campo	**Descripción del Campo**
Nombre de usuario	Nombre de usuario. Sólo se permiten letras en minúscula y dígitos del 0 al 9.
Rol	Rol que va a poseer el usuario que se está creando.
Clave	Contraseña respectiva al usuario.
Reingrese la clave	Aquí se vuelve a escribir la contraseña respectiva a manera de confirmación.
Nombres	Nombres del usuario respectivo.
Apellidos	Apellidos del usuario respectivo.
Identificación	Número de cédula o pasaporte del usuario respectivo
Estado	Estado del usuario.
Botón Guardar	Realiza las validaciones necesarias y guarda los cambios realizados sobre el usuario en la base de datos.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.
Características	
Un usuario con estado inactivo no podrá iniciar sesión en el sistema mientras el administrador no lo active explícitamente.	

3.2.1.3 Opción: Parámetros generales

Esta opción dentro del menú *Sistema* permite al usuario administrador del sistema modificar los valores de los parámetros del sistema.

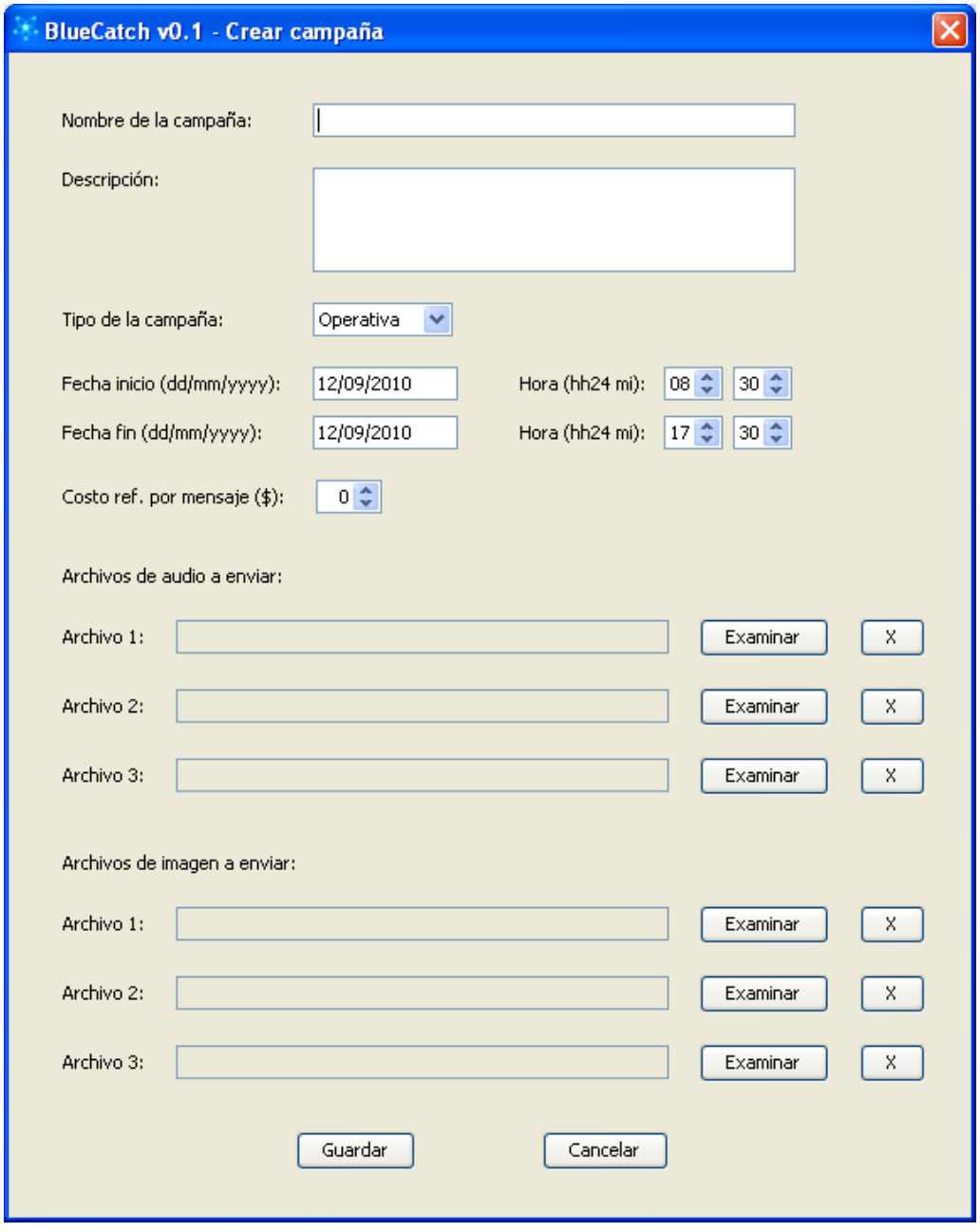


Nombre de Campo	**Descripción del Campo**
Tiempo de bloqueo (min.)	Tiempo en minutos que deberá estar un dispositivo con un estado de bloqueado antes de poder recibir otro archivo de contenido nuevamente. Tiempo de restricción después de cada envío exitoso hacia un dispositivo.
Ruta de reportes exportados	Ruta del disco duro en donde se guardarán los reportes generados por el sistema.
Botón Guardar	Realiza las validaciones necesarias y guarda los cambios realizados sobre los parámetros modificados.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.
Características	
Ambos parámetros son obligatorios.	

3.2.2 Menú: Campaña

3.2.2.1 Opción: Crear campaña

Esta opción dentro del menú *Campaña* permite al usuario administrador del sistema configurar una nueva campaña de marketing, así como sus principales parámetros de envío y ejecución.



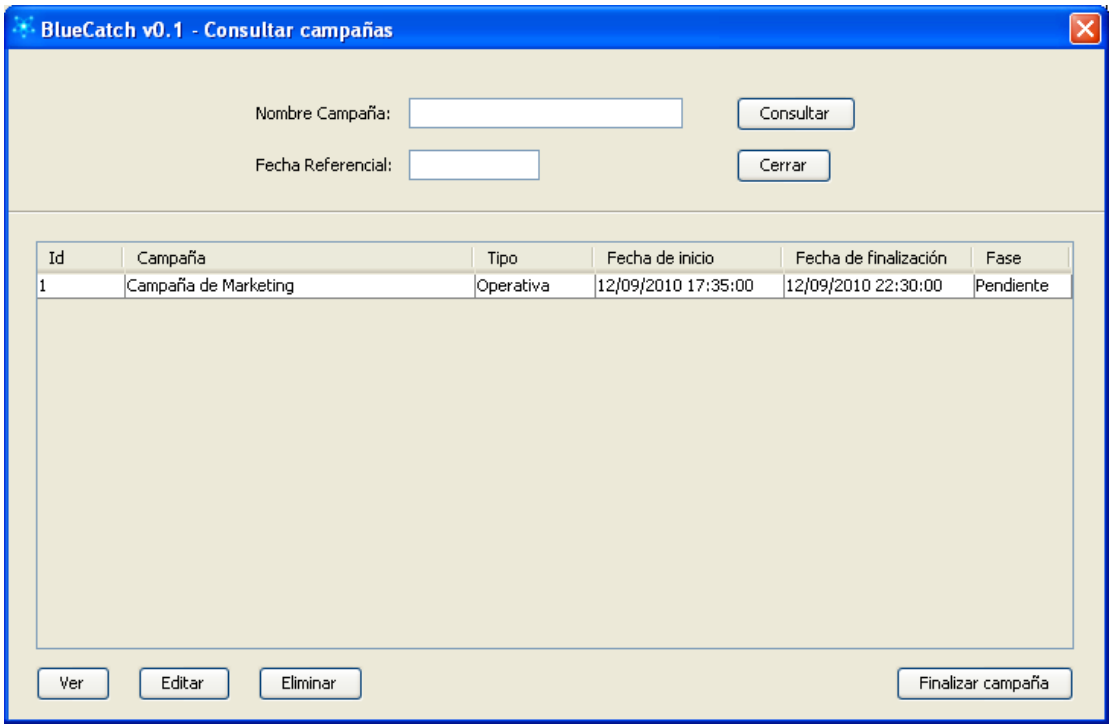
Nombre de Campo	Descripción del Campo
Nombre de la campaña	Nombre que tendrá la nueva campaña que se está configurando.
Descripción	Descripción textual general o detallada acerca de la campaña que se está configurando.
Tipo de la campaña	Tipo de la campaña que se está creando. Puede ser Operativa o Permanente. Una campaña Operativa necesariamente debe tener una fecha de finalización. Por el contrario, una campaña Permanente sólo toma en cuenta la fecha de inicio configurada, ya que deberá ser finalizada explícitamente por el administrador del sistema.
Fecha inicio (dd/mm/yyyy)	Fecha de inicio de la campaña, en el formato especificado.
Hora (hh24 mi)	Hora de inicio de la campaña, en el formato especificado.
Fecha fin (dd/mm/yyyy)	Fecha de finalización de la campaña, en el formato especificado.
Hora (hh24 mi)	Hora de finalización de la campaña, en el formato especificado.
Costo ref. por mensaje ($)	Costo referencial que tendrá cada mensaje de contenido multimedia enviado. Este costo referencial es necesario para poder tener un estimado del costo total de la campaña que se reflejará en el reporte estadístico correspondiente.
Archivos de audio a enviar	En esta sección se agregan uno por uno los archivos de audio (*.mp3) que se desean enviar durante la ejecución de la campaña.
Archivos de imagen a enviar	En esta sección se agregan uno por uno los archivos de imagen (*.jpg) que se desean enviar durante la ejecución de la campaña.
Botón Examinar	Este botón abre un cuadro de selección de archivo. Sirve para escoger cada uno de los archivos a enviar.
Botón X	Este botón elimina la entrada correspondiente en la lista de archivos a enviar.
Botón Guardar	Realiza las validaciones necesarias y guarda la configuración de la campaña creada.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.

Características

No se podrá crear una campaña mientras se encuentre en ejecución una campaña permanente, así como no se podrá crear una campaña permanente mientras existan configuradas campañas operativas que vayan a comenzar luego de la fecha de inicio de la campaña que se está creando.
Como mínimo se deberá configurar un archivo de contenido a enviar, sea de audio o de imagen.

3.2.2.2 Opción: Consultar campañas

Esta opción dentro del menú *Campaña* permite al usuario administrador del sistema consultar las campañas que se encuentran configuradas en el sistema.



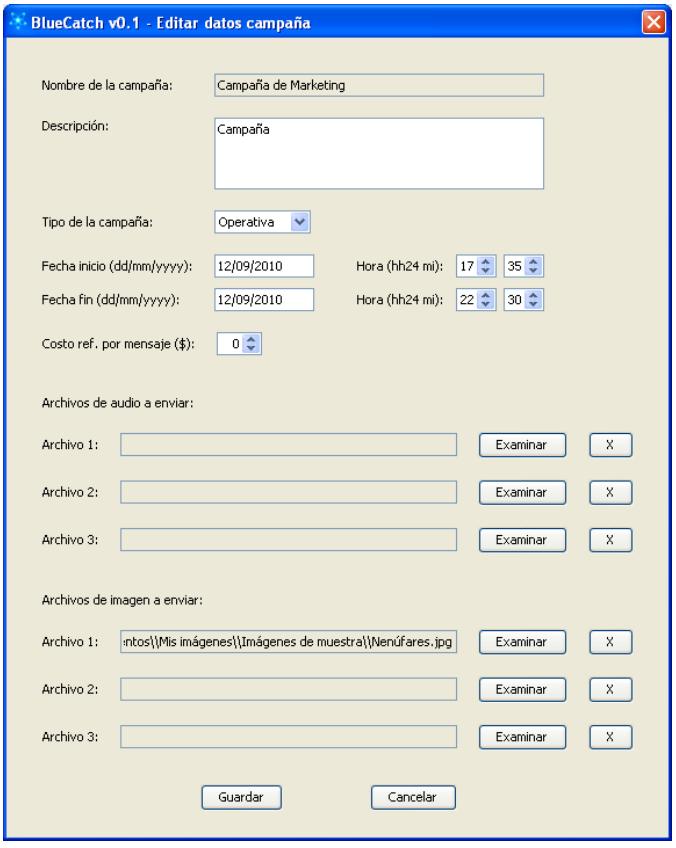
Nombre de Campo	Descripción del Campo
Nombre Campaña	Nombre de la campaña completo o fracción del mismo. Se utiliza para realizar la búsqueda por un patrón.
Fecha Referencial	Fecha referencial de la campaña. Se utilizan para realizar la búsqueda por un patrón.
Id	Código de la campaña.
Campaña	Nombre de la campaña.
Tipo	Tipo de la campaña. Operativa o Permanente.
Fecha Inicio	Fecha de inicio respectiva.
Fecha Fin	Fecha de finalización respectiva.
Fase	Fase en la que se encuentra actualmente la campaña.
Botón Consultar	Realiza la consulta de las campañas de acuerdo a los criterios ingresados.

Botón Cerrar	Cierra la ventana y descarta cualquier cambio realizado.
Botón Ver	Abre una ventana que permite visualizar los datos de la campaña seleccionada.
Botón Editar	Abre una ventana que permite editar los datos de la campaña seleccionada.
Botón Eliminar	Permite eliminar la campaña del sistema, previa confirmación.
Botón Finalizar	Permite finalizar una campaña, guardando como fecha de finalización la fecha actual. Se usa para terminar las campañas permanentes explícitamente.

Características

El campo del formulario de criterios de búsqueda que quede vacío no se tomará en cuenta para realizar la consulta.
No se puede editar una campaña que ya ha iniciado.
No se puede eliminar una campaña que ya ha iniciado.
No se puede finalizar una campaña que no ya ha iniciado

3.2.2.2.1 Opción: Editar configuración de la campaña

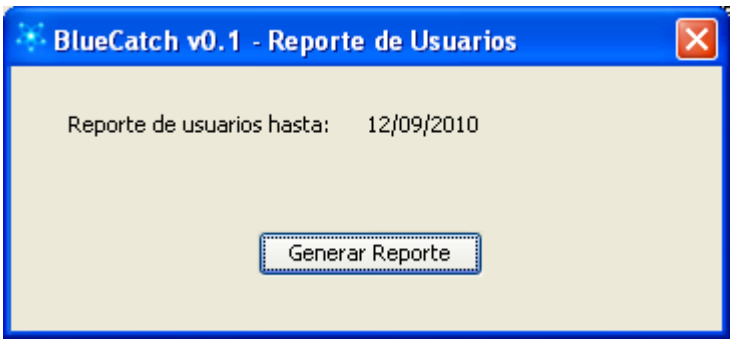


Nombre de Campo	Descripción del Campo
Nombre de la campaña	Nombre que tendrá la nueva campaña que se está configurando.
Descripción	Descripción textual general o detallada acerca de la campaña que se está configurando.
Tipo de la campaña	Tipo de la campaña que se está creando. Puede ser Operativa o Permanente. Una campaña Operativa necesariamente debe tener una fecha de finalización. Por el contrario, una campaña Permanente sólo toma en cuenta la fecha de inicio configurada, ya que deberá ser finalizada explícitamente por el administrador del sistema.
Fecha inicio (dd/mm/yyyy)	Fecha de inicio de la campaña, en el formato especificado.
Hora (hh24 mi)	Hora de inicio de la campaña, en el formato especificado.
Fecha fin (dd/mm/yyyy)	Fecha de finalización de la campaña, en el formato especificado.
Hora (hh24 mi)	Hora de finalización de la campaña, en el formato especificado.
Costo ref. por mensaje ($)	Costo referencial que tendrá cada mensaje de contenido multimedia enviado. Este costo referencial es necesario para poder tener un estimado del costo total de la campaña que se reflejará en el reporte estadístico correspondiente.
Archivos de audio a enviar	En esta sección se agregan uno por uno los archivos de audio (*.mp3) que se desean enviar durante la ejecución de la campaña.
Archivos de imagen a enviar	En esta sección se agregan uno por uno los archivos de imagen (*.jpg) que se desean enviar durante la ejecución de la campaña.
Botón Examinar	Este botón abre un cuadro de selección de archivo. Sirve para escoger cada uno de los archivos a enviar.
Botón X	Este botón elimina la entrada correspondiente en la lista de archivos a enviar.
Botón Guardar	Realiza las validaciones necesarias y guarda la configuración de la campaña creada.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.
Características	
No se podrá cambiar el nombre de la campaña. No se podrá cambiar a una campaña permanente mientras existan configuradas campañas operativas que vayan a comenzar luego de la fecha de inicio de la campaña que se está modificando. Como mínimo se deberá configurar un archivo de contenido a enviar, sea de audio o de imagen.	

3.2.3 Menú: Reportes

3.2.3.1 Opción: Reporte de usuarios

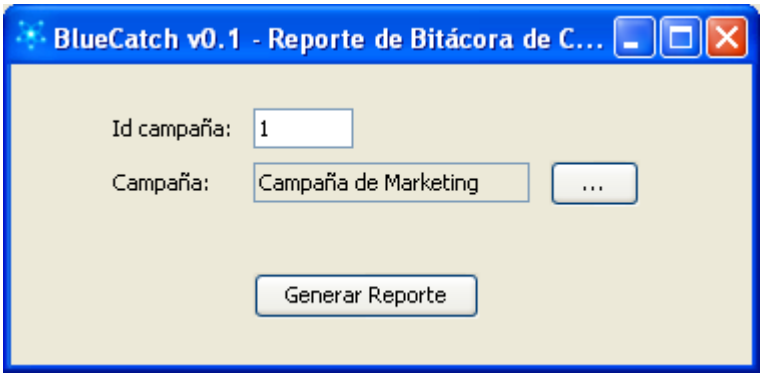
Esta opción dentro del menú *Reportes* permite al usuario administrador del sistema generar un reporte de todos los usuarios que se encuentran ingresados en el sistema, así como también sus respectivos datos personales.



Nombre de Campo	**Descripción del Campo**
Reporte de usuarios hasta	Fecha actual del sistema. Los usuarios que se mostrarán en el reporte serán todos aquellos creados hasta este fecha.
Botón Generar Reporte	Abre la ventana de visualización del reporte, desde donde se podrá imprimir o guardar en algún dispositivo externo.
Características	
No aplica.	

3.2.3.2 Opción: Reporte de Bitácora por Campaña

Esta opción dentro del menú *Reportes* permite al usuario administrador del sistema generar un reporte la bitácora de eventos de transferencia de contenido hacia los dispositivos móviles que han ocurrido durante una determinada campaña.



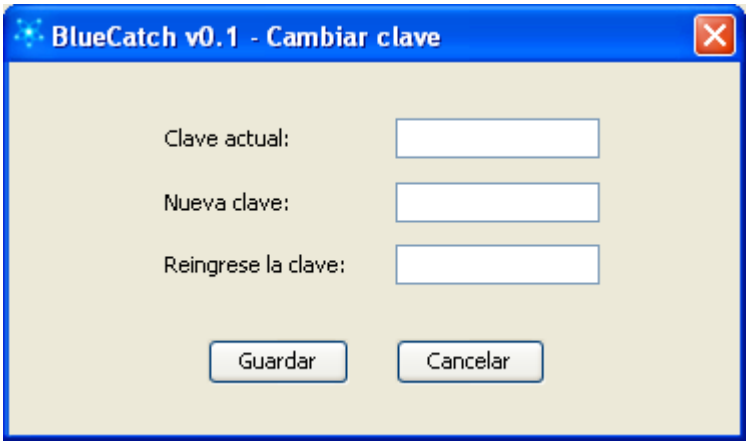
Nombre de Campo	Descripción del Campo
Id campaña	Código de identificación único de la campaña. Se lo puede ingresar manualmente, o buscarlo por medio de la lista de valores
Campaña	Nombre de la campaña seleccionada.
Botón …	Abre una lista de valores con las campañas que existen en el sistema, que permitirá buscar fácilmente la campaña necesaria y seleccionarla.
Botón Generar Reporte	Abre la ventana de visualización del reporte, desde donde se podrá imprimir o guardar en algún dispositivo externo.
Características	
Es necesario escoger una campaña para poder ejecutar el reporte.	

3.3 ROL OPERADOR

3.3.1 Menú: Sistema

3.3.1.1 Opción: Cambiar clave

Esta opción dentro del menú *Sistema* permite al usuario operador del sistema cambiar su clave de ingreso, cambio que se hará efectivo instantáneamente.

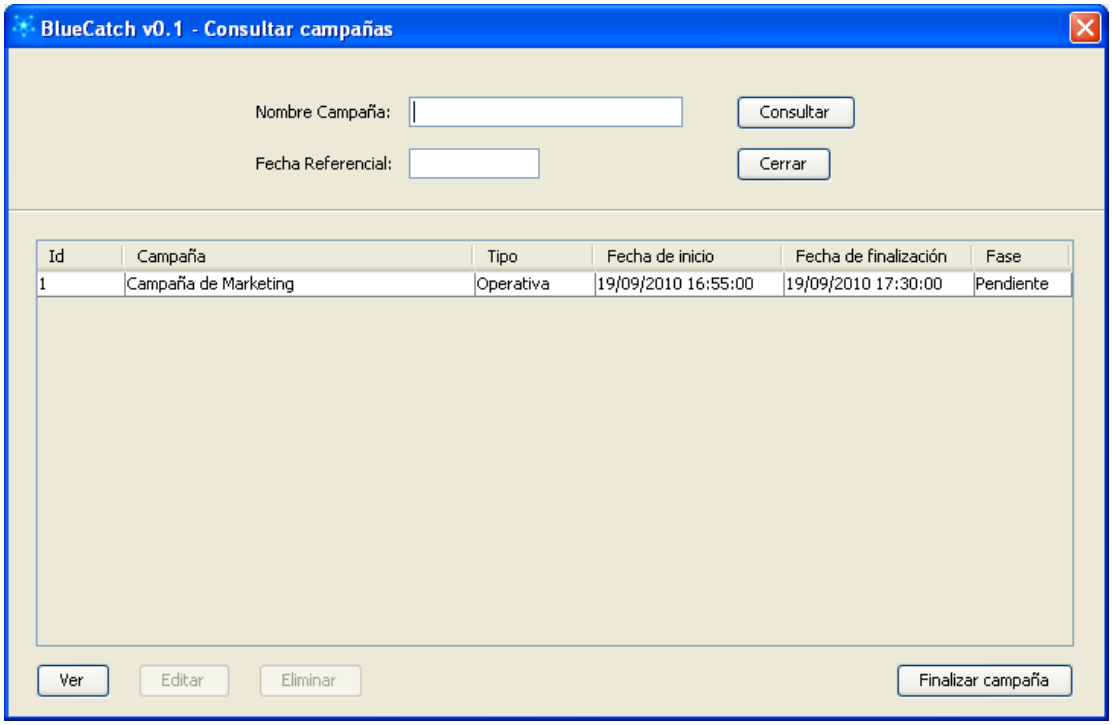


Nombre de Campo	**Descripción del Campo**
Clave actual:	Contraseña respectiva al usuario.
Nueva clave	Nueva contraseña que el usuario desea establecer.
Reingrese la clave	Aquí se vuelve a escribir la contraseña respectiva a manera de confirmación.
Botón Guardar	Realiza las validaciones necesarias y establece la nueva contraseña para el usuario en la base de datos.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.
Características	
No aplica.	

3.3.2 Menú: Campaña

3.3.2.1 Opción: Consultar campañas

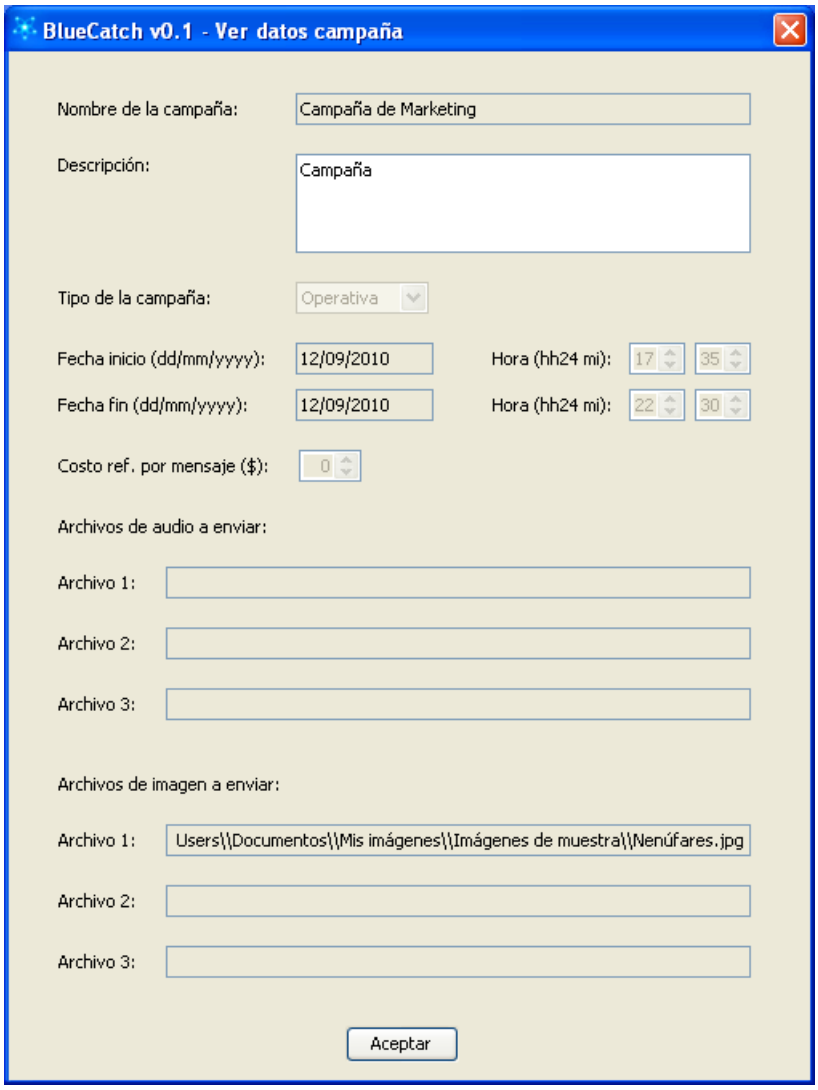
Esta opción dentro del menú *Campaña* permite al usuario operador del sistema consultar las campañas que se encuentran configuradas en el sistema.



Nombre de Campo	**Descripción del Campo**
Nombre Campaña	Nombre de la campaña completo o fracción del mismo. Se utiliza para realizar la búsqueda por un patrón.
Fecha Referencial	Fecha referencial de la campaña. Se utilizan para realizar la búsqueda por un patrón.
Id	Código de la campaña.
Campaña	Nombre de la campaña.
Tipo	Tipo de la campaña. Operativa o Permanente.
Fecha Inicio	Fecha de inicio respectiva.
Fecha Fin	Fecha de finalización respectiva.
Fase	Fase en la que se encuentra actualmente la campaña.
Botón Consultar	Realiza la consulta de las campañas de acuerdo a los criterios ingresados.
Botón Cerrar	Cierra la ventana y descarta cualquier cambio realizado.
Botón Ver	Abre una ventana que permite visualizar los datos de la campaña seleccionada.
Botón Finalizar	Permite finalizar una campaña, guardando como fecha de finalización la fecha actual. Se usa para terminar las campañas permanentes explícitamente.
Características	
El campo del formulario de criterios de búsqueda que quede vacío no se tomará en cuenta para realizar la consulta. No se puede finalizar una campaña que no ya ha iniciado	

3.3.2.1.1 Opción: Ver configuración de la campaña

Permite visualizar el detalle de la configuración de la campaña seleccionada.



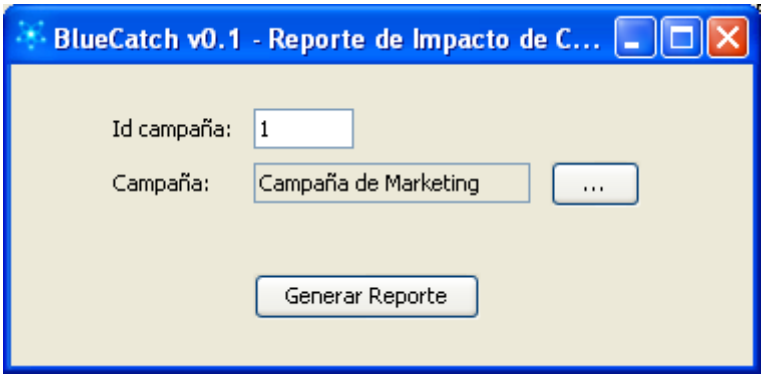
Nombre de Campo	Descripción del Campo
Nombre de la campaña	Nombre que tendrá la nueva campaña que se está configurando.
Descripción	Descripción textual general o detallada acerca de la campaña que se está configurando.
Tipo de la campaña	Tipo de la campaña que se está creando. Puede ser Operativa o Permanente. Una campaña Operativa

	necesariamente debe tener una fecha de finalización. Por el contrario, una campaña Permanente sólo toma en cuenta la fecha de inicio configurada, ya que deberá ser finalizada explícitamente por el administrador del sistema.
Fecha inicio (dd/mm/yyyy)	Fecha de inicio de la campaña, en el formato especificado.
Hora (hh24 mi)	Hora de inicio de la campaña, en el formato especificado.
Fecha fin (dd/mm/yyyy)	Fecha de finalización de la campaña, en el formato especificado.
Hora (hh24 mi)	Hora de finalización de la campaña, en el formato especificado.
Costo ref. por mensaje ($)	Costo referencial que tendrá cada mensaje de contenido multimedia enviado. Este costo referencial es necesario para poder tener un estimado del costo total de la campaña que se reflejará en el reporte estadístico correspondiente.
Archivos de audio a enviar	En esta sección se agregan uno por uno los archivos de audio (*.mp3) que se desean enviar durante la ejecución de la campaña.
Archivos de imagen a enviar	En esta sección se agregan uno por uno los archivos de imagen (*.jpg) que se desean enviar durante la ejecución de la campaña.
Botón Examinar	Este botón abre un cuadro de selección de archivo. Sirve para escoger cada uno de los archivos a enviar.
Botón X	Este botón elimina la entrada correspondiente en la lista de archivos a enviar.
Botón Guardar	Realiza las validaciones necesarias y guarda la configuración de la campaña creada.
Botón Cancelar	Cierra la ventana y descarta cualquier cambio realizado.
Características	
No aplica.	

3.3.3 Menú: Reportes

3.3.3.1 Opción: Reporte de Impacto de Campaña

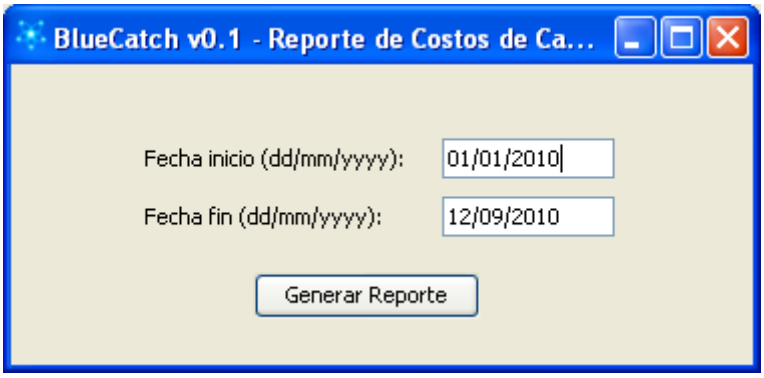
Esta opción dentro del menú *Reportes* permite al usuario operador del sistema generar un reporte que presenta un resumido de la cantidad de transferencias exitosas durante una campaña, permite medir el novel de efectividad de la campaña frente a los clientes por medio de un gráfico circular dividido en secciones.



Nombre de Campo	Descripción del Campo
Id campaña	Código de identificación único de la campaña. Se lo puede ingresar manualmente, o buscarlo por medio de la lista de valores
Campaña	Nombre de la campaña seleccionada.
Botón …	Abre una lista de valores con las campañas que existen en el sistema, que permitirá buscar fácilmente la campaña necesaria y seleccionarla.
Botón Generar Reporte	Abre la ventana de visualización del reporte, desde donde se podrá imprimir o guardar en algún dispositivo externo.
Características	
Es necesario escoger una campaña para poder ejecutar el reporte.	

3.3.3.2 Opción: Reporte de Costos de Campañas

Esta opción dentro del menú *Reportes* permite al usuario operador del sistema generar un reporte que presenta un gráfico de los costos que han tenido las campañas que se han ejecutado en el rango especificado de fechas.



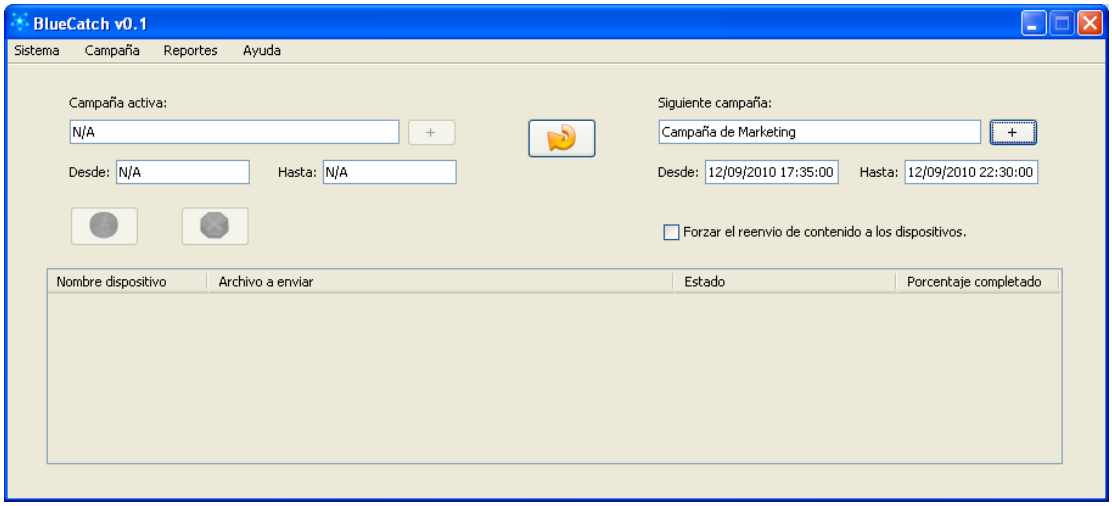
Nombre de Campo	Descripción del Campo
Fecha inicio (dd/mm/yyyy)	Fecha de inicio del rango determinado, en el formato especificado.
Fecha fin (dd/mm/yyyy)	Fecha fin del rango determinado, en el formato especificado.
Botón Generar Reporte	Abre la ventana de visualización del reporte, desde donde se podrá imprimir o guardar en algún dispositivo externo.
Características	
Es necesario escoger una campaña para poder ejecutar el reporte.	

3.4 MONITOR PRINCIPAL DEL SISTEMA

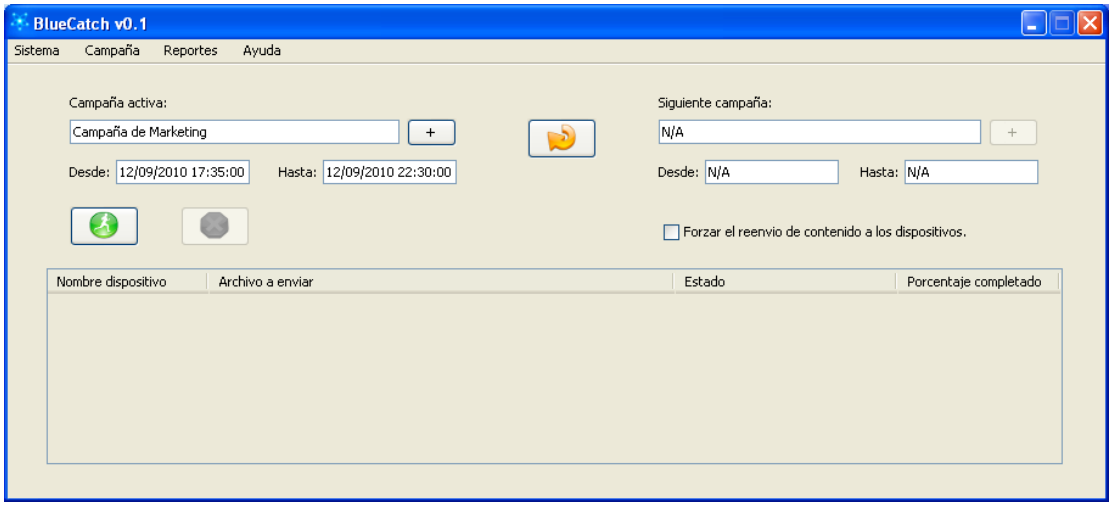
Esta es la pantalla principal del sistema y está habilitada tanto para los usuarios con rol de Operador como para los usuarios con rol de Administrador.

Por medio de ella se podrá comenzar y detener la ejecución de una campaña de marketing previamente configurada. También permite previsualizar los datos de la campaña de marketing más próxima.

Durante la jecución de la campaña, irá mostrando todos los dispositivos que va encontrando así como también los estados correspondientes de cada transeferencia de contenido multimedia hacia cada uno de estos dispositivos.



Una vez que exista una campaña cuya fecha de inicio haya iniciado, se podrá comenzar a correr la ejecución de dicha campaña.

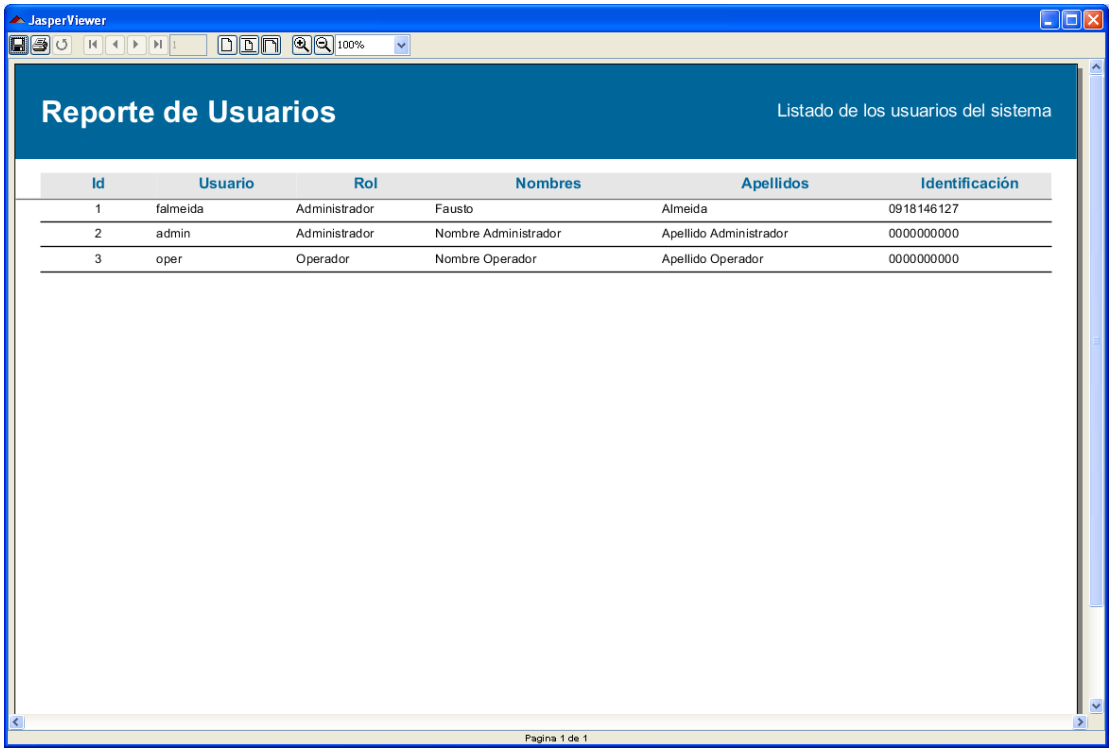


Nombre de Campo	Descripción del Campo
Campaña activa	Nombre de la campaña activa.
Desde	Fecha de inicio de la campaña activa.
Hasta	Fecha de finalización programada para la campaña activa. Si se trata de una campaña de tipo permanente, este campo no aplica.
Siguiente campaña	Nombre de la campaña configurada m;as próxima.
Desde	Fecha de inicio de la campaña configurada m;as próxima.
Hasta	Fecha de finalización programada para la campaña configurada m;as próxima. Si se trata de una campaña de tipo permanente, este campo no aplica.
CheckBox Forzar reenvío	Si está activado, forza el reenvio de contenido multimedia a los dispositivos sin tomar en cuenta si poseen o no un estado bloqueado.
Botón Refrescar	Vuelve a cargar las campañas en el monitor, actualizando su ubicación, ya sea en el panel de Campaña Activa, o en el panel de Siguiente Campaña.
Botón Iniciar campaña	Inicia el proceso de envío de contenido para la campaña activa.
Botón Detener campaña	Pausa el proceso de envío de contenido para la campaña activa.
Características	
No aplica.	

4 REPORTES

4.1 REPORTES ADMINISTRATIVOS

4.1.1 Reporte de usuarios



Nombre de Campo	**Descripción del Campo**
Id	Código del usuario.
Usuario	Nombre de usuario.
Rol	Rol que posee el usuario.
Nombres	Nombres del usuario respectivo.
Apellidos	Apellidos del usuario respectivo.
Identificación	Número de cédula o pasaporte del usuario respectivo.
Características	
No aplica.	

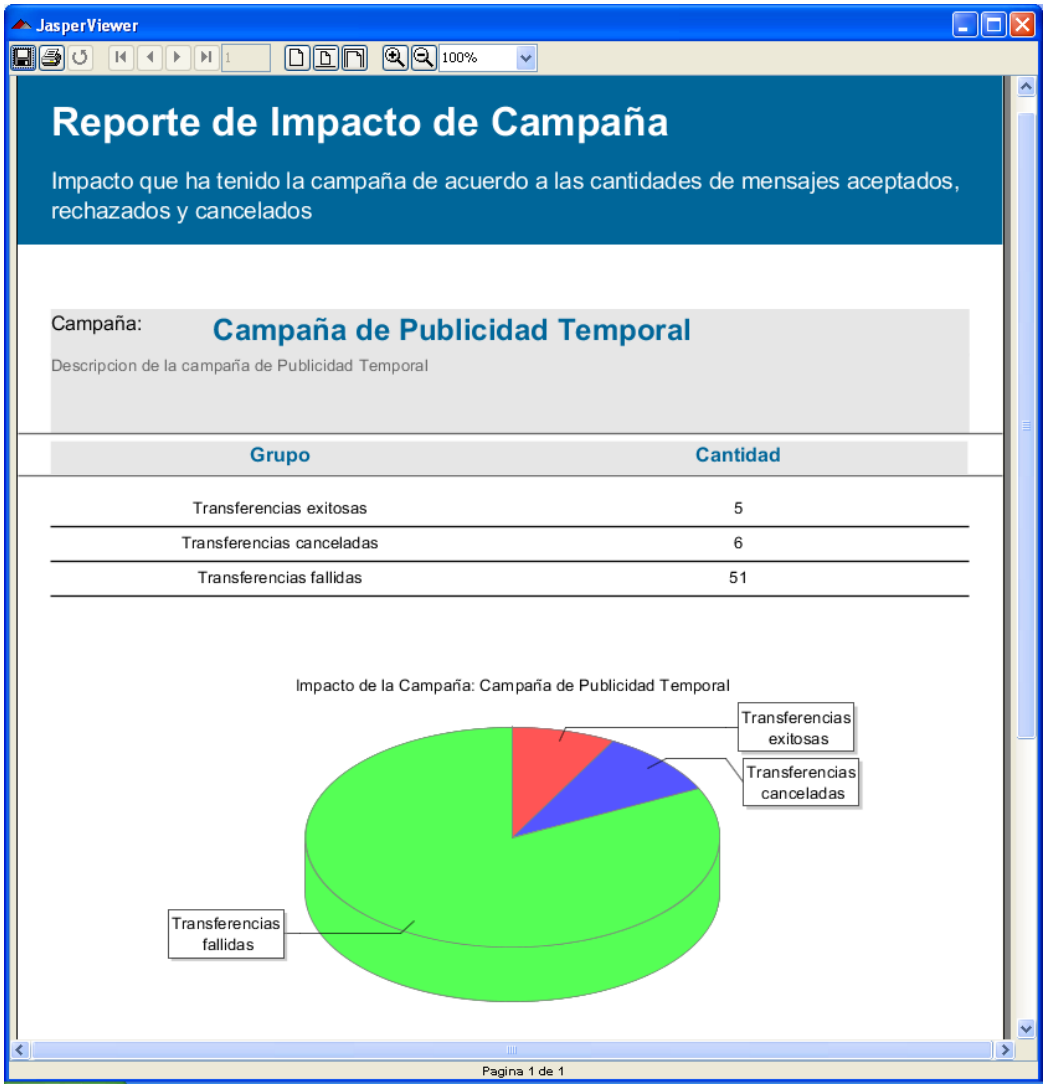
4.1.2 Reporte de Bitácora de Campaña

JasperViewer
Reporte de Bitácora de Campaña
Eventos generados durante la transmisión de los archivos a los clientes
Campaña
Campaña Java
Campaña Java
Fecha de registro
Nombre del dispositivo
Dirección
Observación
Usuario
Archivo de contenido:
C:\Documents and Settings\Administrador\Mis documentos\NetBeansProjects\JABlueCatch\Otros\Bender.mp3
16/08/2010 14:05:24 Aby A8F274B9F909 Dispositivo detectado admin
16/08/2010 14:05:33 Aby A8F274B9F909 Inicio transmisión admin
16/08/2010 14:06:23 Aby A8F274B9F909 Transmisión exitosa admin
12/08/2010 01:30:24 Fas N97 mini 5C57C83C81DC Dispositivo detectado admin
12/08/2010 01:30:35 Fas N97 mini 5C57C83C81DC Inicio transmisión admin
12/08/2010 01:30:59 Fas N97 mini 5C57C83C81DC Transmisión exitosa admin
15/08/2010 15:11:45 Fas N97 mini 5C57C83C81DC Dispositivo detectado admin
15/08/2010 15:11:55 Fas N97 mini 5C57C83C81DC Inicio transmisión admin
15/08/2010 15:12:03 Fas N97 mini 5C57C83C81DC Transmisión exitosa admin
16/08/2010 14:05:24 Fas N97 mini 5C57C83C81DC Dispositivo detectado admin
16/08/2010 14:05:35 Fas N97 mini 5C57C83C81DC Inicio transmisión admin
16/08/2010 14:05:48 Fas N97 mini 5C57C83C81DC Transmisión exitosa admin
15/08/2010 15:12:45 LG GM210 0025E5B78493 Dispositivo detectado admin
15/08/2010 15:12:47 LG GM210 0025E5B78493 No se ha encontrado el servicio en el dispositivo 0025E5B78493. Envío fallido. admin
15/08/2010 15:13:06 LG GM210 0025E5B78493 Dispositivo detectado admin
Pagina 1 de 2

Nombre de Campo	**Descripción del Campo**
Campaña	Nombre de la campaña correspondiente.
Fecha Registro	Fecha y hora en que se sucitó el evento de bitácora.
Nombre Dispositivo	Nombre del dispositivo móvil sobre el cual ocurrió el evento.
Dirección	Dirección Bluetooth única del dispositivo móvil sobre el cual ocurrió el evento de bitácora.
Observación	Observación del evento de bitácora.
Usuario	Usuario en cuya sesión ocurrió el evento de bitácora.
Archivo de contenido	Ruta y nombre del archivo de contenido acerca de cuya transferencia se registró el evento.
Características	
A continuación de cada descripción de archivo de contenido, se mostrarán todos los registros de eventos de bitácora correspondientes.	

4.2 REPORTES OPERATIVOS

4.2.1 Reporte de Impacto de Campaña



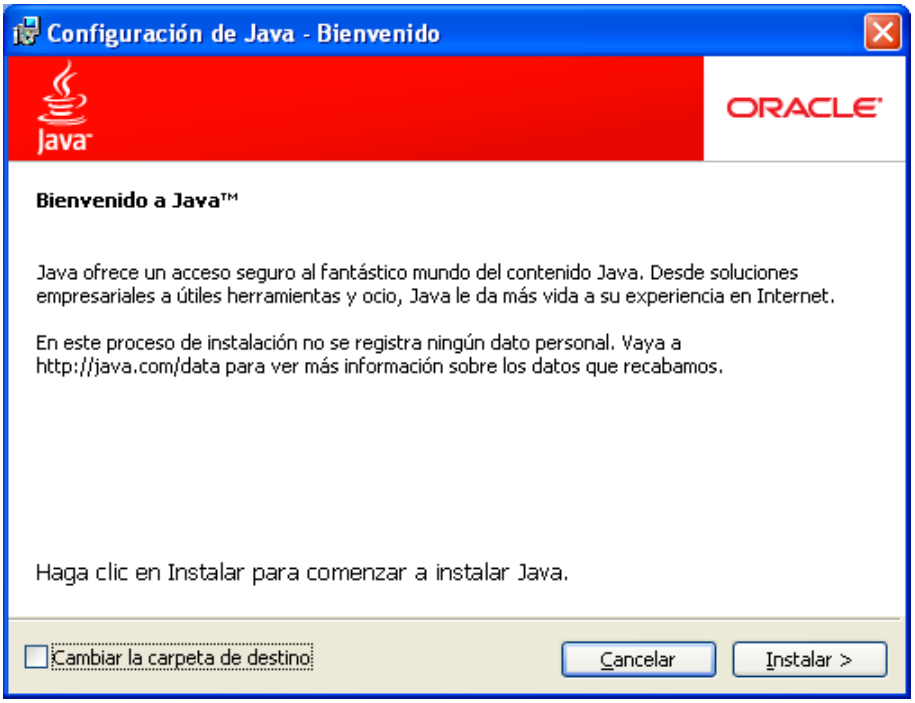
Nombre de Campo	Descripción del Campo
Campaña	Nombre y descripción de la campaña correspondiente.
Grupo	Descripción de cada agrupación por tipo de transferencia.
Cantidad	Cantidad de trasferencias de cada agrupación.
Características	
Este reporte presenta un gráfico circular que permite demostrar de una manera más amigable el resultado del impacto que ha tenido la campaña sobre los clientes.	

4.2.2 Reporte de Costos de Campañas

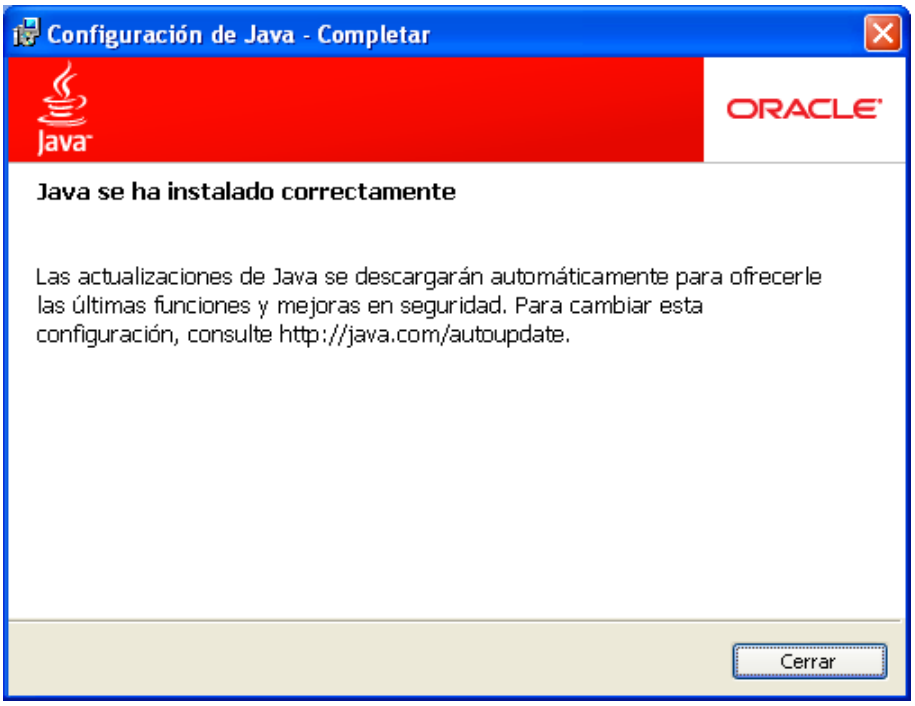
Nombre de Campo	Descripción del Campo
Campaña	Nombre de la campaña correspondiente.
Descripción	Descripción de la campaña correspondiente.
Fecha de inicio	Fecha de inicio de la campaña correspondiente.
Fecha de finalización	Fecha de finalización de la campaña correspondiente.
Mensajes enviados	Número total de mensajes enviados exitosamente durante la campaña correspondiente.
Costo mensaje	Costo referencial en dólares que tiene cada mensaje en la campaña correspondiente.
Total	Total de costo de la campaña correspondiente.
Características	
Este reporte presenta un gráfico vectorial que permite demostrar de una manera más precisa la variación de los costos de la campañas a tavés del tiempo.	

5 ANEXOS

5.1 INSTALACIÓN DE JAVA RUNTIME ENVIROMENT

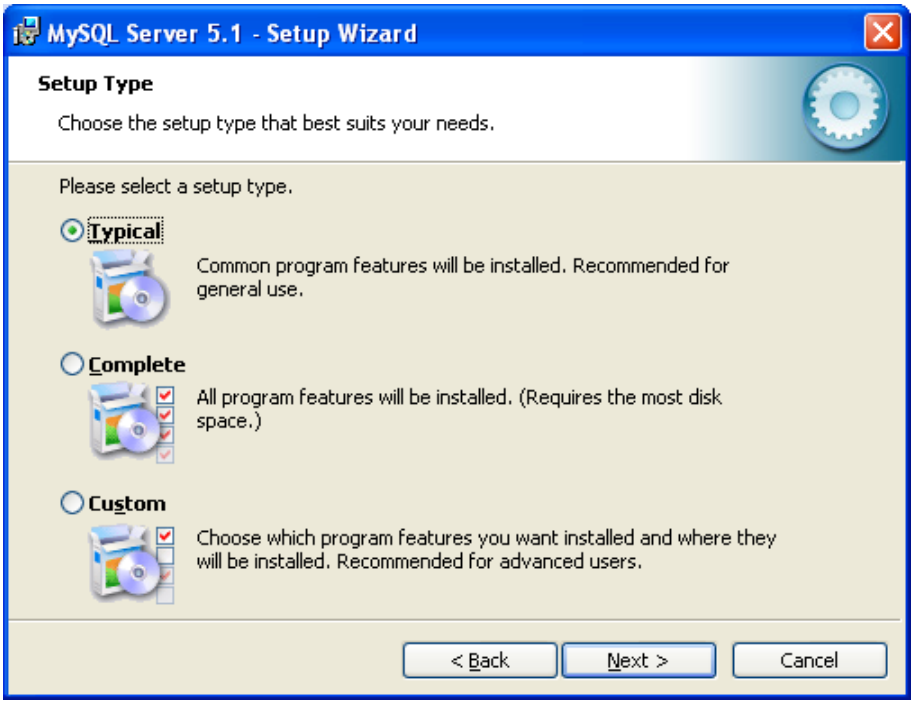

Configuración de Java - Bienvenido
Java
ORACLE
Bienvenido a Java™
Java ofrece un acceso seguro al fantástico mundo del contenido Java. Desde soluciones empresariales a útiles herramientas y ocio, Java le da más vida a su experiencia en Internet.
En este proceso de instalación no se registra ningún dato personal. Vaya a http://java.com/data para ver más información sobre los datos que recabamos.
Haga clic en Instalar para comenzar a instalar Java.
Cambiar la carpeta de destino
Cancelar
Instalar >

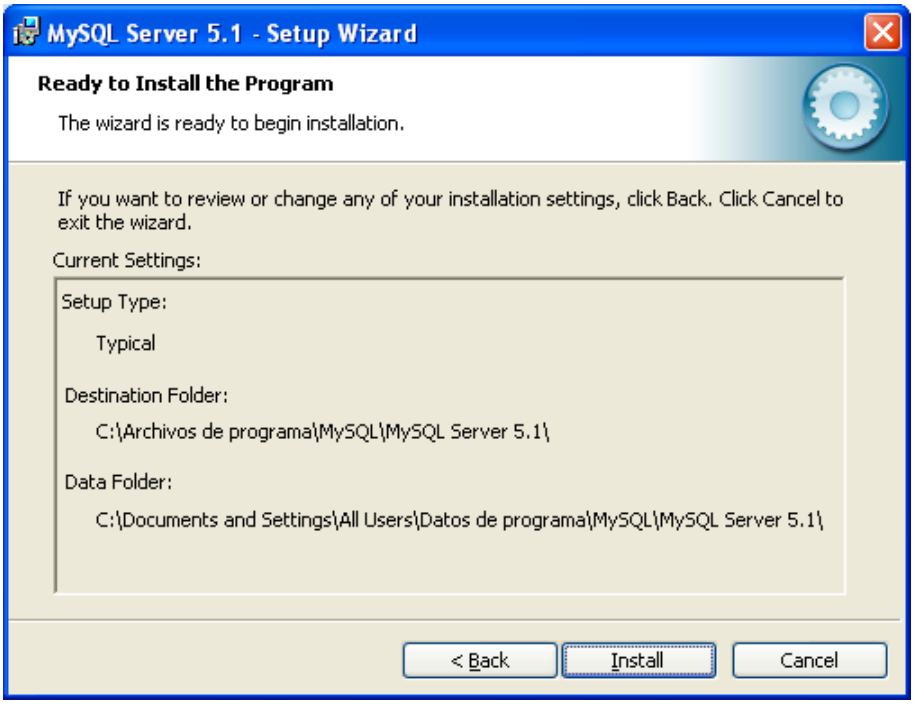
Configuración de Java - Progreso
Instalando Java
Este proceso puede tardar unos minutos.
Java
ORACLE
Estado: Instalar el entorno Java Runtime Environment
Ya puede contar con un conjunto de aplicaciones completo y gratuito compatible con Microsoft Office
Conjunto de aplicaciones de procesamiento de texto, hojas de cálculo, presentaciones, dibujo y bases de datos
Permite leer, editar y guardar archivos de Microsoft Office
Disponible en más de 70 idiomas y compatible con los sistemas operativos Solaris, Windows, Linux y Mac
Utiliza el formato de archivo abierto OpenDocument como formato predeterminado
Exportación a PDF con un solo clic
Writer
Calc
Impress
Base
Draw
Math
OpenOffice.org

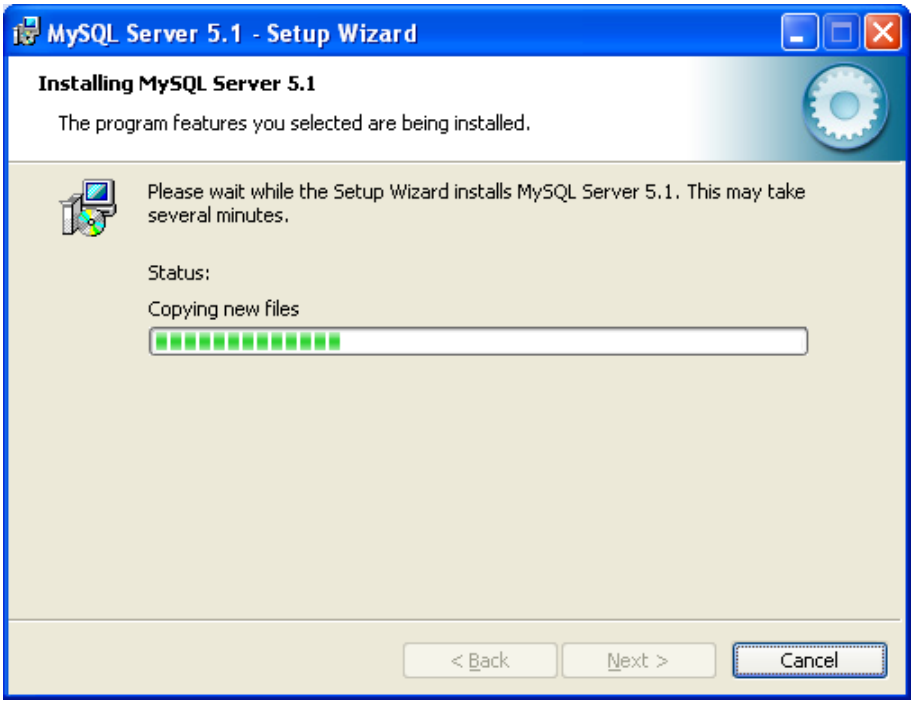

Configuración de Java - Completar
Java
ORACLE
Java se ha instalado correctamente
Las actualizaciones de Java se descargarán automáticamente para ofrecerle las últimas funciones y mejoras en seguridad. Para cambiar esta configuración, consulte http://java.com/autoupdate.
Cerrar

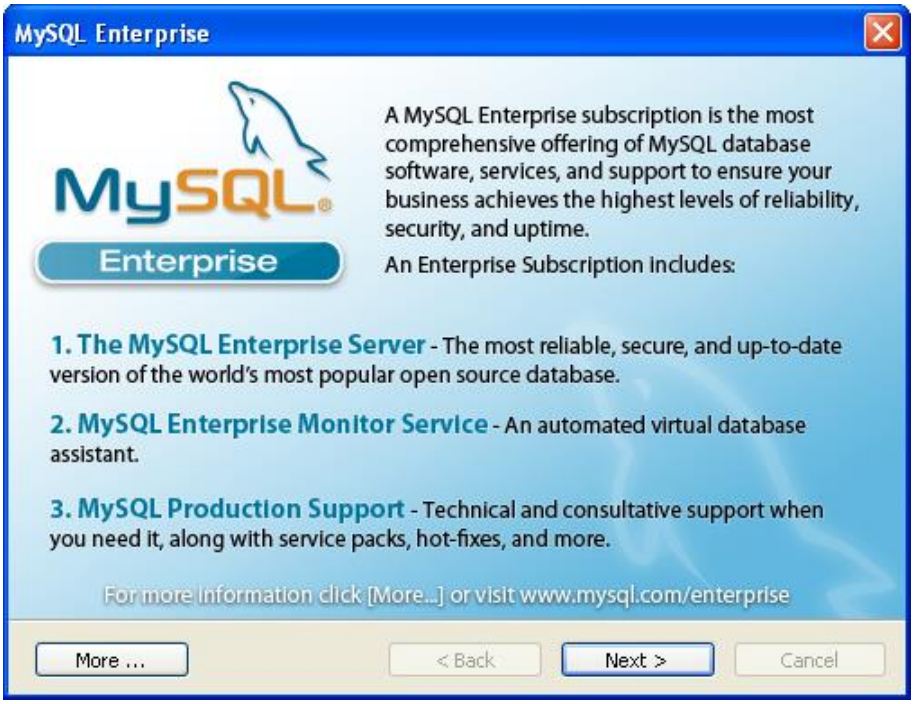
5.2 INSTALACIÓN DE BASE DE DATOS MYSQL

MySQL Server 5.1 - Setup Wizard
Welcome to the Setup Wizard for MySQL Server 5.1
The Setup Wizard will install MySQL Server 5.1 release 5.1.46 on your computer. To continue, click Next.
WARNING: This program is protected by copyright law.
MySQL
< Back
Next >
Cancel


MySQL Server 5.1 - Setup Wizard
Setup Type
Choose the setup type that best suits your needs.
Please select a setup type.
Typical
Common program features will be installed. Recommended for general use.
Complete
All program features will be installed. (Requires the most disk space.)
Custom
Choose which program features you want installed and where they will be installed. Recommended for advanced users.
< Back
Next >
Cancel

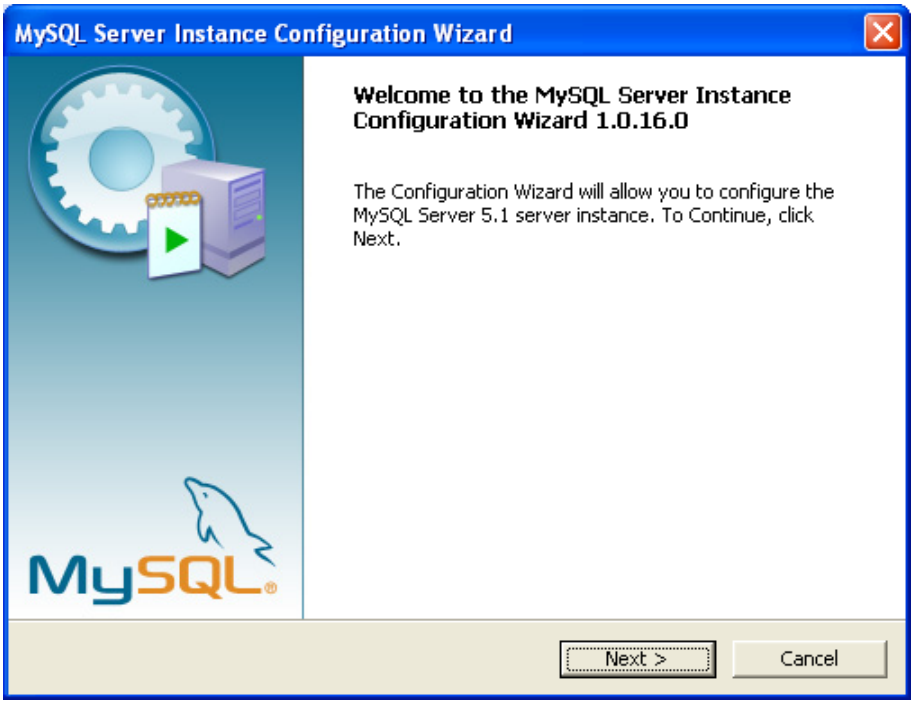

MySQL Server 5.1 - Setup Wizard
Ready to Install the Program
The wizard is ready to begin installation.
If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.
Current Settings:
Setup Type:
Typical
Destination Folder:
C:\Archivos de programa\MySQL\MySQL Server 5.1\
Data Folder:
C:\Documents and Settings\All Users\Datos de programa\MySQL\MySQL Server 5.1\
< Back
Install
Cancel

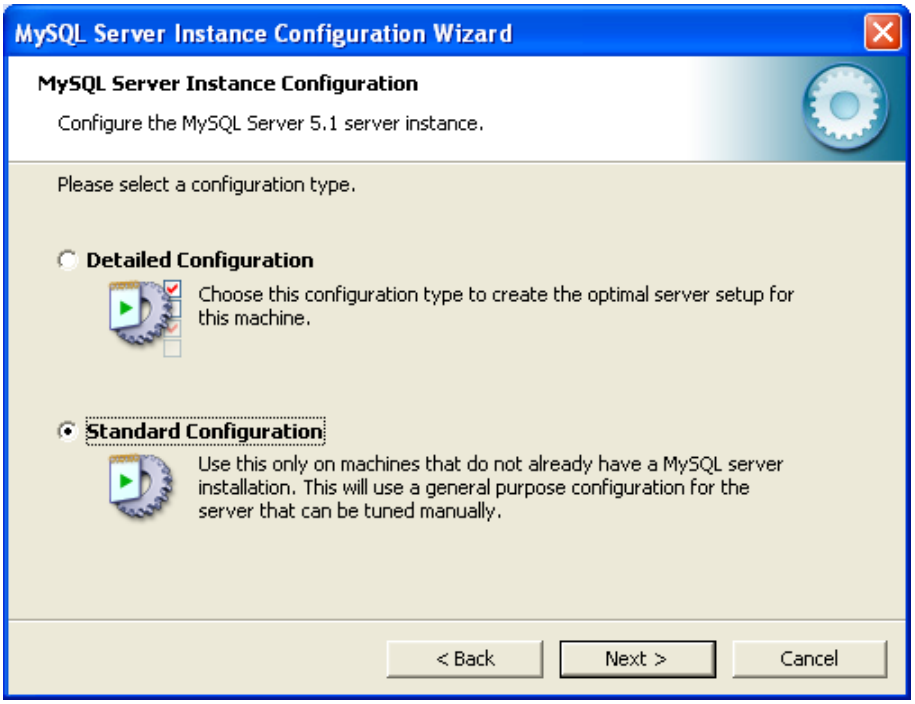

MySQL Server 5.1 - Setup Wizard
Installing MySQL Server 5.1
The program features you selected are being installed.
Please wait while the Setup Wizard installs MySQL Server 5.1. This may take several minutes.
Status:
Copying new files
< Back
Next >
Cancel

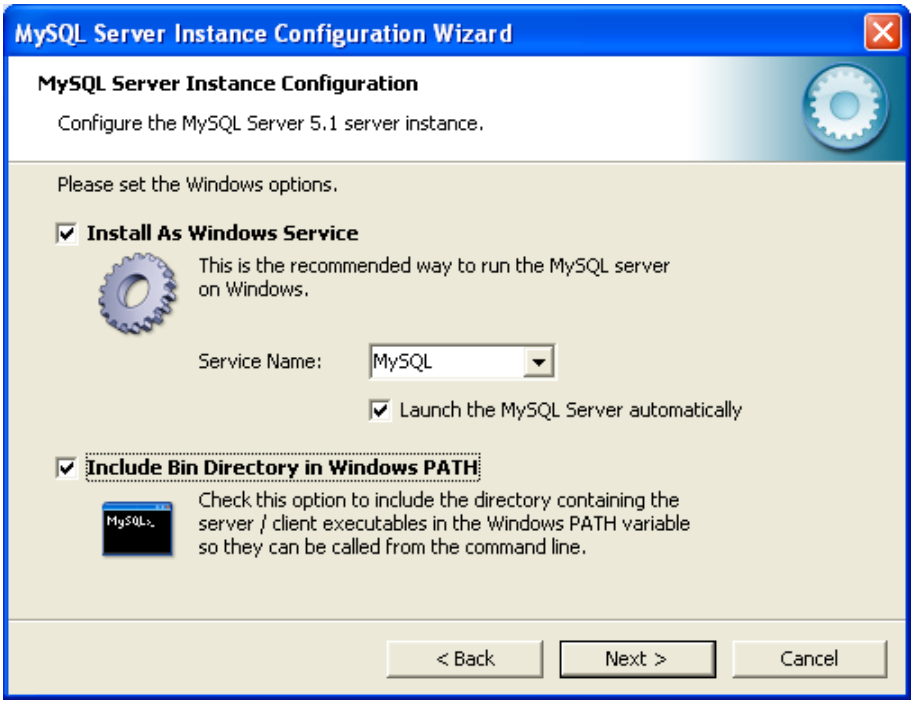

MySQL Enterprise
MySQL
Enterprise
A MySQL Enterprise subscription is the most comprehensive offering of MySQL database software, services, and support to ensure your business achieves the highest levels of reliability, security, and uptime.
An Enterprise Subscription includes:
1. The MySQL Enterprise Server - The most reliable, secure, and up-to-date version of the world's most popular open source database.
2. MySQL Enterprise Monitor Service - An automated virtual database assistant.
3. MySQL Production Support - Technical and consultative support when you need it, along with service packs, hot-fixes, and more.
For more information click [More...] or visit www.mysql.com/enterprise
More ...
< Back
Next >
Cancel

MySQL Enterprise
MySQL
Enterprise
The MySQL Enterprise Monitor Service
• Quickly identifies your most expensive SQL code across all your servers.
• MySQL Advisors and 125+ Best Practice Rules ensure security and performance.
• Alerts and Expert Advice on how to fix problems and tune for peak performance.
For more information click [More...] or visit www.mysql.com/enterprise
More ...
< Back
Next >
Cancel

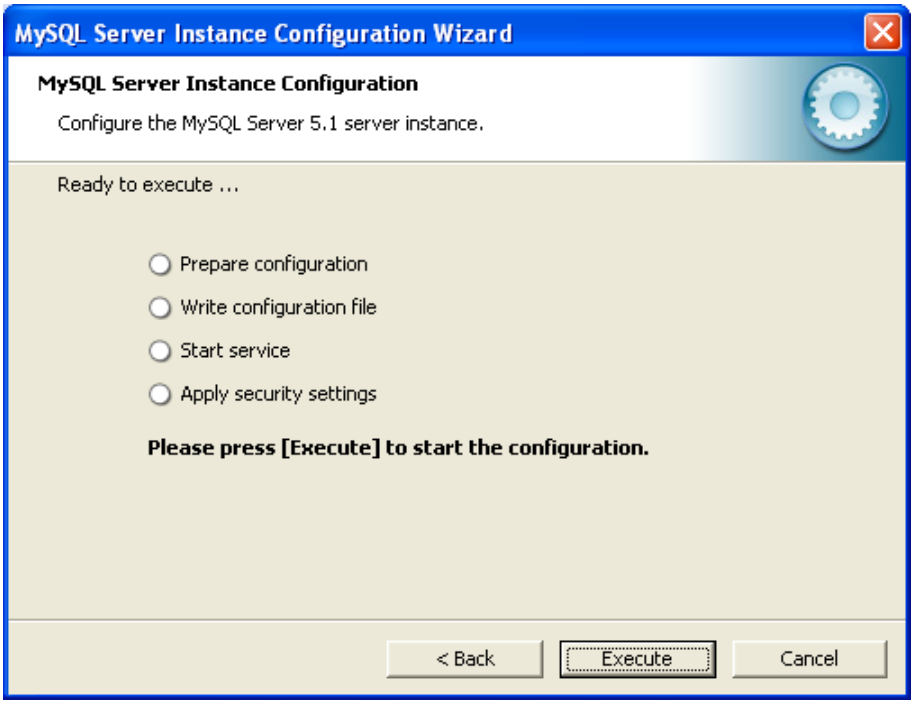
MySQL Server 5.1 - Setup Wizard
Wizard Completed
Setup has finished installing MySQL Server 5.1. Click Finish to exit the wizard.
Configure the MySQL Server now
Use this option to generate an optimized MySQL config file, setup a Windows service running on a dedicated port and to set the password for the root account.
Register the MySQL Server now
Use this option to register this MySQL server with SunConnect service, to receive automatic update notifications on future releases and other free offerings. A browser window will open briefly to allow you to complete the registration.
MySQL
< Back
Finish
Cancel

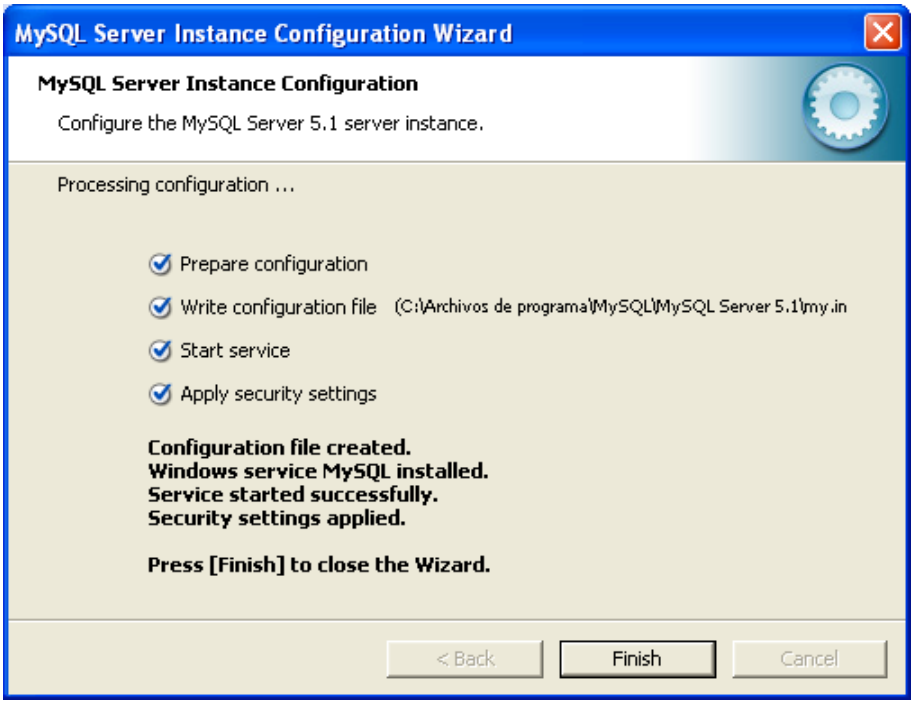

MySQL Server Instance Configuration Wizard
Welcome to the MySQL Server Instance Configuration Wizard 1.0.16.0
The Configuration Wizard will allow you to configure the MySQL Server 5.1 server instance. To Continue, click Next.
MySQL
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.1 server instance.
Please select a configuration type.
Detailed Configuration
Choose this configuration type to create the optimal server setup for this machine.
Standard Configuration
Use this only on machines that do not already have a MySQL server installation. This will use a general purpose configuration for the server that can be tuned manually.
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.1 server instance.
Please set the Windows options.
Install As Windows Service
This is the recommended way to run the MySQL server on Windows.
Service Name:
MySQL
Launch the MySQL Server automatically
Include Bin Directory in Windows PATH
Check this option to include the directory containing the server / client executables in the Windows PATH variable so they can be called from the command line.
< Back
Next >
Cancel

MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.1 server instance.
Please set the security options.
Modify Security Settings
root
New root password:
******
Enter the root password.
Confirm:
******
Retype the password.
Enable root access from remote machines
Create An Anonymous Account
This option will create an anonymous account on this server. Please note that this can lead to an insecure system.
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.1 server instance.
Ready to execute ...
Prepare configuration
Write configuration file
Start service
Apply security settings
Please press [Execute] to start the configuration.
< Back
Execute
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.1 server instance.
Processing configuration ...
Prepare configuration
Write configuration file (C:\Archivos de programa\MySQL\MySQL Server 5.1\my.in
Start service
Apply security settings
Configuration file created.
Windows service MySQL installed.
Service started successfully.
Security settings applied.
Press [Finish] to close the Wizard.
< Back
Finish
Cancel